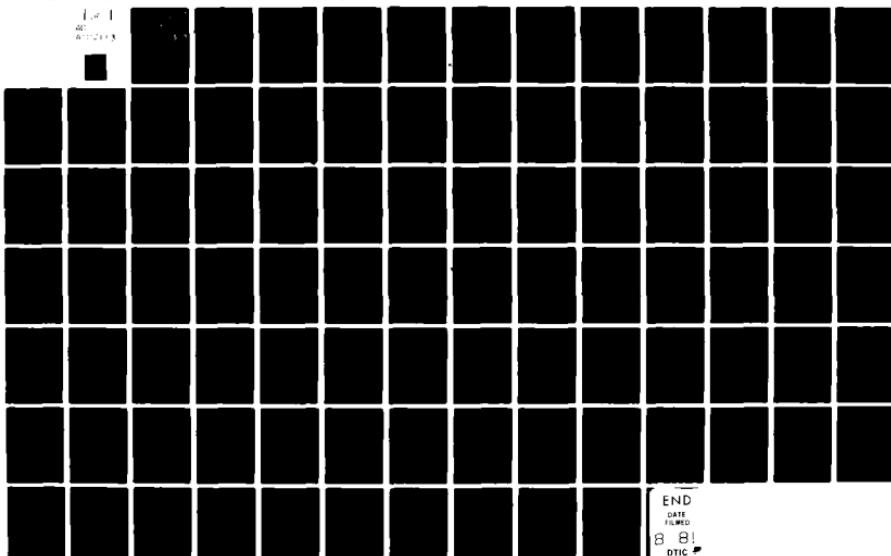


AD-A102 113    NAVAL AIR DEVELOPMENT CENTER WARMINSTER PA    SOFTWARE --ETC    F/G 17/2  
HIGH SPEED MULTIPLEX BUS PROTOCOL STUDY. (U)  
JUN 80    J S VERDI

UNCLASSIFIED    NADC-81049-50

NL



END  
DATE  
FILED  
8-81  
DTIC

REPORT NO. NADC-81049-50

LEVEL II (12)

AD A102113



HIGH SPEED MULTIPLEX BUS  
PROTOCOL STUDY

DTIC ELECTED  
S JUL 27 1981

James S. Verdi  
Software and Computer Directorate  
NAVAL AIR DEVELOPMENT CENTER  
Warminster, Pennsylvania 18974

15 June 1980

PHASE REPORT  
Airtask No. A03A360G/001B/1F21200000  
Work Unit No. WF21.241.091/ZD111

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

Prepared for:  
NAVAL AIR SYSTEMS COMMAND  
Department of the Navy  
Washington, DC 20361

DTIC FILE COPY

81 7 27 103

## N O T I C E S

REPORT NUMBERING SYSTEM - The numbering of technical project reports issued by the Naval Air Development Center is arranged for specific identification purposes. Each number consists of the Center acronym, the calendar year in which the number was assigned, the sequence number of the report within the specific calendar year, and the official 2-digit correspondence code of the Command Office or the Functional Directorate responsible for the report. For example: Report No. NADC-78015-20 indicates the fifteenth Center report for the year 1978, and prepared by the Systems Directorate. The numerical codes are as follows:

CODE	OFFICE OR DIRECTORATE
00	Commander, Naval Air Development Center
01	Technical Director, Naval Air Development Center
02	Comptroller
10	Directorate Command Projects
20	Systems Directorate
30	Sensors & Avionics Technology Directorate
40	Communication & Navigation Technology Directorate
50	Software Computer Directorate
60	Aircraft & Crew Systems Technology Directorate
70	Planning Assessment Resources
80	Engineering Support Group

PRODUCT ENDORSEMENT - The discussion or instructions concerning commercial products herein do not constitute an endorsement by the Government nor do they convey or imply the license or right to use such products.

APPROVED BY: R. J. Mitchell

DATE: 6/24/81

(17) WF 115,00000  
WF 21241094

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS  
BEFORE COMPLETING FORM

1. REPORT NUMBER NADC-81049-50	2. GOVT ACCESSION NO. AD-A402 443	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and subtitle) High Speed Multiplex Bus Protocol Study	5. TYPE OF REPORT & PERIOD COVERED 9 Phase Report	
7. AUTHOR(s) James S. Verdi	8. CONTRACT OR GRANT NUMBER(s) <i>(16) F21200 F21244</i>	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Software and Computer Directorate (Code 5022) Naval Air Development Center Warminster, PA 18974	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS AIRTASK A03A360G/001B/ 1F21200000 W.U. NO. WF21.241.091/ZD111	
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Air Development Center Software and Computer Directorate (Code 5022) Warminster, PA 18974	12. REPORT DATE 15 June 1980 <i>(12) 92</i>	13. NUMBER OF PAGES 82
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Air Systems Command (AIR-360) Department of the Navy Washington, DC 20361	15. SECURITY CLASS. (of this report) Unclassified	
16. DISTRIBUTION STATEMENT (of this Report)  Approval for public release; distribution unlimited	17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)	
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Avionics Bus Bus Protocol Digital Multiplex Bus Data Busses	Command/Response Priority Bus Access Advanced Digital Bus	Bus Protocol Functions Data Multiplexing Communication Protocols
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report presents the current results of an evaluation performed in support of the development of an advanced high speed digital avionics data bus. Emphasis is placed on certain areas of protocol design and implementation which have not been approached very aggressively in other development efforts. These areas include a transaction oriented protocol design and greatly increased level of protocol definition stressing VLSI (and perhaps VHSIC) implementation. A detailed but preliminary protocol design is included to illustrate the concepts presented.		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 68 IS OBSOLETE  
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

393538

✓  
✓  
✓

TABLE OF CONTENTS

		<u>PAGE NO.</u>
1.0	GENERAL	1
1.1	Scope	1
1.2	Objectives	1
2.0	BACKGROUND	2
2.1	Evolutionary Aspects	2
2.1.1	Impact on Distributed Processing Aspects	2
2.1.2	Modularization	4
2.2	Applicable Documents	4
3.0	PROTOCOL DESCRIPTION	5
3.1	Definitions	5
3.1.1	Block	5
3.1.2	Message	5
3.1.3	Transaction	5
3.1.4	Link (Linkage)	5
3.1.5	Chain	6
3.1.6	Remote Terminal (RT) Address	6
3.1.7	Subaddress	6
3.1.8	Command Word (CW)	9
3.1.9	Status Word (SW)	9
3.1.10	Data Word (DW)	9
3.1.11	Control Data Word (CDW)	9
3.1.12	Normal Data Word (NDW)	9
3.1.13	Interrupt Data Word (IDW)	9
3.1.14	Mode Discrete	10
3.1.15	Nesting	10
3.1.16	Suggest	10
3.1.17	Request	11
3.1.18	Poll Request Sequence (PRS)	11
3.1.19	Mode Discrete Sequence (MDS)	11
3.1.20	Information Transfer Request (ITR)	11
3.1.21	Information Transfer Suggest (ITS)	11
3.1.22	Bus Controller (BC)	12
3.1.23	Remote Terminal	12
3.1.24	Command Frame	12
3.1.25	Status Frame	12
3.1.26	Bus Interface Unit (BIU)	12
3.1.27	Extended Command Word (ECW)	13
3.1.28	Extended Status Word (ESW)	13
3.1.29	Bus Control Data	13
3.1.30	Check Word	14

TABLE OF CONTENTS

	<u>PAGE NO.</u>
3.2 Word Formats	14
3.2.1 General	14
3.2.1.1 Bit Priority	14
3.2.1.2 Word Types	15
3.2.2 Command Frame	15
3.2.2.1 Command Word (CW)	15
3.2.2.1.1 Mode Discrete (CC = 0)	15
3.2.2.1.1.1 Dynamic Bus Control Offer (CA = 0)	18
3.2.2.1.1.2 Transmit Status Word (CA = 1)	18
3.2.2.1.1.3 Initiate Self Test (CA = 2)	18
3.2.2.1.1.4 Transmit Bit Message (CA = 3)	18
3.2.2.1.1.5 Transmit Status Words (CA = 4)	18
3.2.2.1.1.6 Reset (CA = 6)	19
3.2.2.1.1.7 Transmitter Shutdown (CA = 7)	19
3.2.2.1.1.8 Transmitter Shutdown Override (CA = 8)	19
3.2.2.1.1.9 Transmit Last Command Frame (CA = 9)	19
3.2.2.1.1.10 Receive Bus Control Data (CA = 10)	19
3.2.2.1.1.11 Transmit Bus Control Data (CA = 11)	20
3.2.2.1.1.12 Unconditional Halt-All Currently Active Sequences (CA = 12)	20
3.2.2.1.1.13 Unconditional Halt - the Currently Active (CA = 13)	20
3.2.2.1.1.14 Loop Test (CA = 15)	20
3.2.2.1.2 Transmit Command Code (CC = 1)	20
3.2.2.1.2.1 Error Retry/Initial (Bit #3)	21
3.2.2.1.2.2 Request/Suggest (Bit #2)	21
3.2.2.1.2.3 Chained/Unchained (Bit #1)	21
3.2.2.1.2.4 RT-RT/Normal (Bit #0)	21
3.2.2.1.3 Receive Command Code (CC = 2)	21
3.2.2.1.3.1 Error Retry/Initial (Bit #3)	22
3.2.2.1.3.2 Request/Suggest (Bit #2)	22
3.2.2.1.3.3 Chained/Unchained (Bit #1)	22
3.2.2.1.3.4 RT-RT/Normal (Bit #0)	22
3.2.2.2 CW+1	22
3.2.2.2.1 Source Information Field	22
3.2.2.2.2 Extended Command Amplifier	24
3.2.2.2.2.1 Nest Field	24
3.2.2.2.2.2 Block Type	24
3.2.2.2.2.3 Sequence Type	24
3.2.2.3 Block and Residual Word Count	25
3.2.2.4 Source/Destination Word	25
3.2.2.5 Frame Check ECW	25
3.2.3 Status Frame	25
3.2.3.1 Status Word (SW)	26
3.2.3.1.1 Idle/MDS Response (SC = 0)	26
3.2.3.1.1.1 Bus Control Acceptance (SA = 0)	26
3.2.3.1.1.2 Idle; No Requests/Suggests (SA = 1)	29

TABLE OF CONTENTS

	<u>PAGE NO.</u>
3.2.3.1.1.3 BIT Executing (SA = 2)	29
3.2.3.1.1.4 BIT Message Follows (SA = 3)	29
3.2.3.1.1.5 Busy; Message Lost (SA = 4)	29
3.2.3.1.1.6 Busy; Not Ready Yet (SA = 5)	29
3.2.3.1.1.7 Reset Complete (SA = 6)	29
3.2.3.1.1.8 Transmitter Disabled (SA = 7)	30
3.2.3.1.1.9 Transmitter Enabled (SA = 8)	30
3.2.3.1.1.10 Last Command Frame Follows (SA = 9)	30
3.2.3.1.1.11 Bus Control Data Received (SA = 10)	30
3.2.3.1.1.12 Bus Control Data Follows (SA = 11)	30
3.2.3.1.1.13 Unconditional Halt - All Sequences (SA = 12)	30
3.2.3.1.1.14 Unconditional Halt - Currently Active Sequence (SA = 13)	31
3.2.3.1.1.15 Unconditional Halt Received (SA = 14)	31
3.2.3.1.2 Transmit (SC = 1)	31
3.2.3.1.2.1 Error Retry/Initial (Bit #3)	31
3.2.3.1.2.2 Service (Transaction) Request (Bit #2)	31
3.2.3.1.2.3 Chained/Unchained (Bit #1)	32
3.2.3.1.2.4 RT-RT/Normal (Bit #0)	32
3.2.3.1.3 Receive (SC = 2)	32
3.2.3.1.3.1 Error Retry/Initial (Bit #3)	32
3.2.3.1.3.2 Service (Transaction) Request (Bit #2)	33
3.2.3.1.3.3 Chained/Unchained (Bit #1)	33
3.2.3.1.3.4 RT-RT/Normal (Bit #0)	33
3.2.3.1.4 Error Condition (SC = 3)	33
3.2.3.2 SW+1	35
3.2.3.2.1 Source Information Field	35
3.2.3.2.2 Extended Status Amplifier (ESA)	35
3.2.3.2.2.2 Nest Field	35
3.2.3.2.2.2 Block Type	35
3.2.3.2.2.3 Sequence Type	36
3.2.3.3 Block and Residual Word Count ESW	36
3.2.3.4 Source/Destination ESW	36
3.2.3.5 Frame Check ESW	36
3.3 Basic Operating Modes	36
3.3.1 Bus Controller (BC) Mode	37
3.3.2 Remote Terminal (RT) Mode	37
3.3.3 RT-RT Mode - General	37
3.3.3.1 Chained RT-RT Transaction	38
3.3.3.2 Unchained RT-RT Transaction	38
3.3.3.3 Broadcast RT-RT Transactions	38
3.3.4 Broadcast Operation	46
3.3.4.1 Time Out (T.O.) Mechanism	47
3.3.4.2 Class/Priority Allocation and the T.O. Mechanism	47

TABLE OF CONTENTS

	<u>PAGE NO.</u>
3.3.4.3 Polling and Broadcast	50
3.3.4.4 Data Block Transfers and Broadcast	50
3.3.4.5 Dynamic Bus Allocation and Broadcast	50
3.3.5 Error Modes	55
3.3.5.1 Error Types	58
3.3.5.1.1 Protocol Errors	58
3.3.5.5.1.1.1 Illegal Subaddress	58
3.3.5.1.1.2 Illegal Mode Discrete	58
3.3.5.1.1.3 Subsystem Error	58
3.3.5.1.1.4 Illegal Sequence	58
3.3.5.1.2 Message Error	58
3.3.5.1.2.1 Word Count Error	59
3.3.5.1.2.2 Message Length Error	59
3.3.5.1.2.3 Sync Waveform/Bit Count Error	59
3.3.5.1.2.4 Parity Error	59
3.3.5.1.2.5 FCW Error	59
3.3.5.1.2.6 DCW Error	59
3.3.5.1.2.7 Time Out Response Error	60
3.3.5.2 Error Recovery	60
3.4 Control and Information Exchange Sequences	60
3.4.1 Mode Discrete Sequence (MDS)	63
3.4.2 Poll Request Sequence (PRS)	63
3.4.3 Information Transfer Request (ITR)	63
3.4.3.1 Transmit Request (TxR)	63
3.4.3.2 Receive Request (RxR)	63
3.4.4 Information Transfer Suggest (ITS)	63
3.4.4.1 Transmit Suggest (TxS)	63
3.4.4.2 Receive Suggest (RxS)	66
3.4.5 Nesting	66
3.4.5.1 Bus Unit (RT) Initiated Nest	66
3.4.5.2 Bus Controller (BC) Initiated Nest	66
3.4.5.3 RT-RT Nested Transactions	66
3.4.6 Loop Test	72
3.4.6.1 Loop Test Detailed Description	72
4.0 GENERAL BUS CHARACTERISTICS	74
4.1 Bus Architecture	74
4.2 Message Continuity	74
4.3 Transmission Media	74

TABLE OF CONTENTS

	<u>PAGE NO.</u>
4.4	74
4.4.1	74
4.4.2	74
4.4.3	74
4.5	75
4.6	75
5.0 CONCLUSIONS	78
5.1 Application Considerations I	78
5.2 Application Considerations II	78
BIBLIOGRAPHY	80
ABBREVIATIONS AND ACRONYMS	81

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Avail and/or	
Dist	Special
A	

List of Figures

	<u>PAGE NO.</u>	
2.1-1	Subsystem I/O Processing Functional Partitioning	3
3.1.7-1	Multiple Bus Unit Implementation of RT Address and Subaddress	7
3.1.7-2	Alternative Multiple Bus Unit Implementation of RT Address and Subaddress	8
3.2.2-1	Command Frame Content	16
3.2.2.1-1	Command Word Definition	17
3.2.2.2-1	CW+1 - Source Information and Extended Command Amplifier (ECA)	23
3.2.3-1	Status Frame Content	27
3.2.3.1-1	Status Word Definition	28
3.2.3.2-1	SW+1 - Source Information and Extended Status Amplifier (ESA)	34
3.3.3-1	Chained RT-RT Transaction	39-41
3.3.3-2	Unchained RT-RT Transaction	42-45
3.3.4.1-1	High Level Bus State Transition Diagram; BC-RT Poll or Transaction Request	48
3.3.4.1-2	Response Time Out Interval Measurement	49
3.3.4.3-1	Broadcast Poll	51-52
3.3.4.4-1	Broadcast RxS Message	53-54
3.3.4.5-1	Dynamic Bus Control Allocation (DBCA)	56-57
3.4-1	General Message Types	61
3.4-2	General Message Types	62
3.4.3-1	High Level BC State Transition Diagram - BC-RT Transaction Request	64
3.4.3-2	High Level RT State Transition Diagram - BC-RT Transfer Request	65

NADC-81049-50

List of Figures

	<u>PAGE NO.</u>
3.4.5.1-1      Nested Transaction - RT Initiated	67-68
3.4.5.2-1      BC Initiated Nested Transaction	69-70
3.4.5.3-1      Nested Transactions Involving RT-RT Transfers	71
3.4.5.3-2      RT-RT Nested Transaction	73
4.5-1           Command Frame Word Identifiers	76
4.5-2           Status Frame Word Identifiers	77

## 1.0 GENERAL

This document was produced in accordance with planned milestones of the Information Handling, High Speed Multiplex Bus Task effort at NADC, code 5022, under block R&D funding from NAVAIR 360B.

### 1.1 Scope

The scope of this effort is the evaluation of the avionics data processing environment from the point of view of interprocessor and general intersubsystem communications. The scope of this document includes the definition of communication sequences and rules which will enhance bus communications capabilities and lead to the optimization of a generally applicable protocol for avionics systems.

### 1.2 Objective

This document is a result of an ongoing effort to develop a digital high speed multiplex data bus concept in sufficient detail to be a useful tool for application to the avionics environment. A further objective of this effort is to provide ideas and insights which may be applied to the problem of developing the techniques and criteria which would lead to a standard bus realization at a most cost effective and interoperable level across all Navy avionics platforms.

## 2.0 BACKGROUND

### 2.1 Evolutionary Aspects

#### 2.1.1 Impact on Distribution Processing Concepts

The evolution of avionics systems has been accelerated by the implementation of the data bus concept in military airborne applications. By virtue of data bussing, the advances of LSI technology and the impetus of space exploration, the development of distributed processing systems for avionics applications is being further encouraged. It seems very appropriate now, after so much has been learned from multiplex data bus development efforts to date, that with the trend towards distributed processing of system functions further effort should be expended towards detailed definition and partitioning of communication control functions within the host subsystem. Consistent with this philosophy, this document presents three basic design goals plus the technical approach which integrates them into an advanced avionics bus concept. Briefly, these three goals are:

- a) Layered approach to protocol definition emphasizing peer level communications.
- b) Significantly increased detail of protocol definition over other military/industry/academic approaches (section 2.1.2).
- c) Compatibility with, but extension of, MIL-STD-1553 formats and concepts (section 2.1.2)

The layered approach to protocol definition is adopted in an effort to minimize complexity (and therefore cost) in interface design, to minimize the impact and cost of changes in interface implementation, and to promote the gradual standardization of protocol definition in an orderly fashion. This last point emphasizes the requirement for interoperability of the advanced bus across many platforms.

There is significant precedent for adopting a layered, peer level communications approach. The International Organization for Standardization (ISO) emphasizes this concept in the "Reference Model of Open System Interconnection". Within the scope of digital avionics bussing applications, these ideas are entirely applicable.

The generalized communications functions, which have thus far been identified and partitioned as bus controller or remote terminal functions as opposed to host or application functions, are indicated in figure 2.1-1 and detailed in the later sections of this document. The functional allocation shown in figure 2.1-1 seems to be suitable for applications which involve a distributed processing system architecture. This document applies to the Link Level and Communication Sequence Control Level of figure 2.1-1.

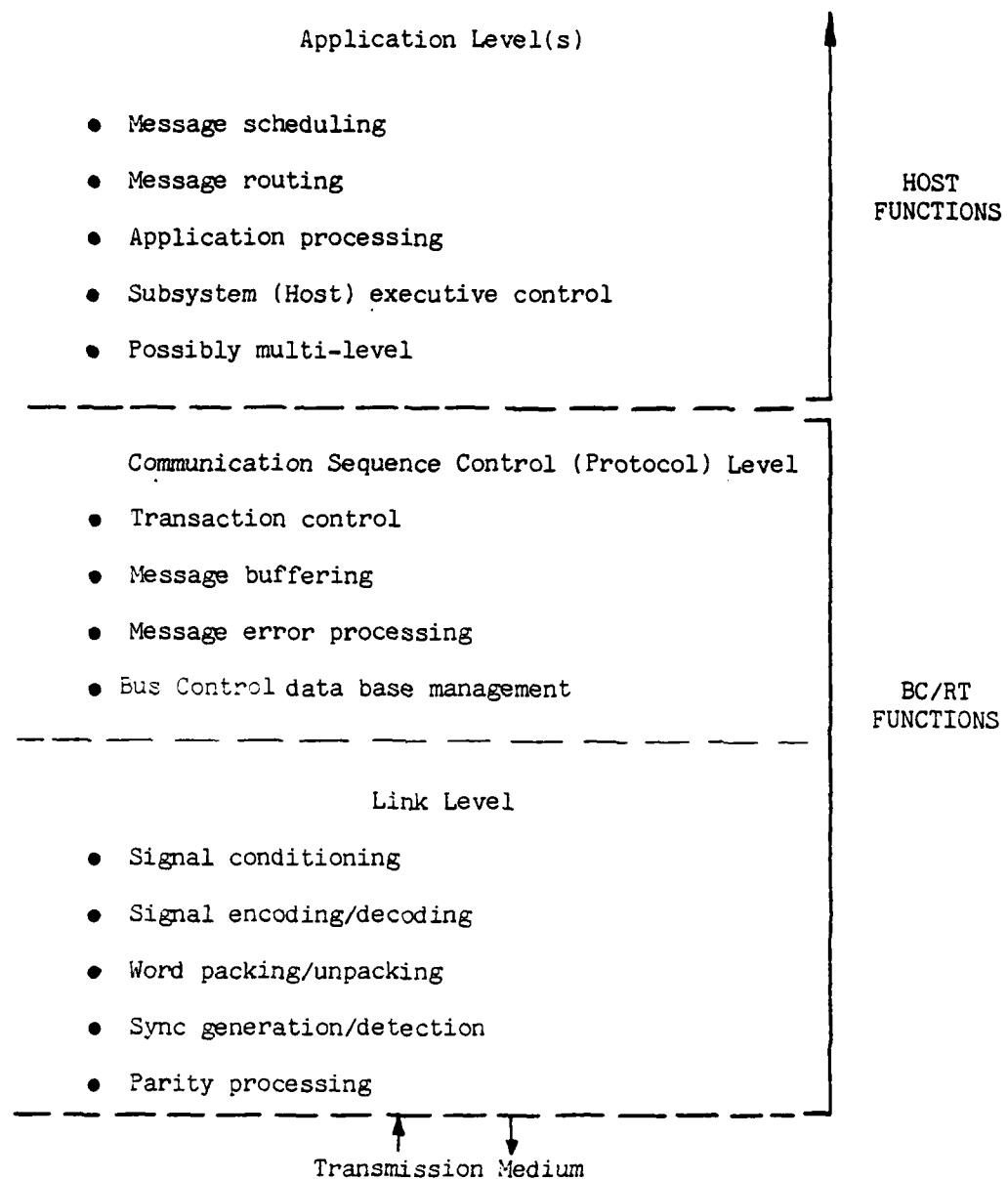


FIGURE 2.1-1 Subsystem I/O Processing Functional Partitioning

### 2.1.2 Modularization

The development of MIL-STD-1553 and the current product lines of Circuit Technology, Inc., (CTI) and Harris Semiconductor (to name just two) in the bus technology area, have demonstrated that an appropriate level of definition can lead to specific, dedicated, modularized control functions. This effort, in conjunction with ongoing developments in advance integrated circuit fabrication techniques and in the Local Computer Network (LCN) communications technology area, encourages further refinement of bus protocol definition so as to promote interoperability of an advanced bus to a much higher level than is possible with MIL-STD-1553. Specifically, we project that VHSIC (Very High Speed Integrated Circuit) developments in integrated circuit speed and functional density will provide capabilities which may be adopted for this application. Consistent with these trends and using MIL-STD-1553 as a baseline, it is felt that an advanced high speed interprocessor data bus can be developed which would incorporate the layered (peer level communications) approach discussed in the previous section, along with the detailed protocol definition and modularization concepts discussed in this section. Interoperability, up to but not including the subsystem level, is the intended goal.

This report contains a preliminary version of the High Speed Multiplex Bus (HSMB) protocol design. It is included in its present form (further work is required as yet in several areas) in order to illustrate in a substantial manner the relatively high order bus control functions which are projected for implementation in a VLSI/VHSIC chip or chip set. It is expected that through careful analysis and selection of functions which are to be implemented in high speed, high density hardware the following two major results will be achieved: (1) the interoperability of the basic bus communications subsystem will be promoted across all avionics platforms to a much greater extent than is currently achieved; and (2) system software development and maintenance costs will be reduced for all systems due to the expected reduction in programming requirements in the I/O (Input/Output) area. The HSMB protocol makes use of MIL-STD-1553B functions as a subset but goes beyond to completely specify a broadcast mode of operation and a more detailed implementation method for dynamic bus control allocation. Also included are enhanced error detection features, provisions for interrupt capabilities where necessary, and features which are designed to ease integration with a host subsystem. The layered, peer-level communications approach is stressed in this report; however, the preliminary HSMB protocol document (to be made available as a stand alone document in late FY81) will reflect the structure of the "Reference Model of Open Systems Interconnection" in a much more formal manner. Additionally, there are certain areas of the protocol which will be defined in more detail plus other areas that may require modification to a limited extent. These areas include (but are not necessarily limited to) the Bus Control Data Block (section 3.1.2.9), synchronization waveform (figures 4.5-1 and 4.5-2), and Loop Test definition (section 3.4.6).

### 2.2 Applicable Documents

(See Bibliography)

### 3.0 PROTOCOL DESCRIPTION

This section and its subsections present in detail the description of the proposed protocol. Included are all required word formats, modes of operation, information exchange rules and legal message sequences. Error detection and error recovery are also addressed.

#### 3.1 Definitions

##### 3.1.1 Block

A block may be referred to as either a data block consisting of from one to 1024(1k) 32 bit words or a message block which consists of a command or status frame plus a data block. With respect to the count word in the command or status frame, the Block Count refers to the number of full data blocks of 1024 32-bit words.

##### 3.1.2 Message

A message consists of an information transfer (control and/or data) from one bus unit to another bus unit plus a return information transfer involving a status response. A transfer is considered a one way transmission to or from a bus unit. Therefore, a message consists of two information transfers. In the context of "transmitting a message", "receiving a message", "multiple messages", etc., it is implied that a status response is associated with each message reference.

##### 3.1.3 Transaction

A transaction is the complete exchange of information between bus units regardless of how many messages that are required to accomplish the exchange. A transaction may consist of one message (as in a poll or a one block information exchange) or it may consist of many messages (as with a multiblock data exchange). A transaction includes any and all Request messages which are necessary to effect a data exchange.

##### 3.1.4 Link (Linkage)

When there is a transfer of information between two units on the bus (e.g., when a transaction is underway) a linkage is said to exist between the units. There are two possible linkages which may exist; chained or unchained. These are defined in the next subsection.

### 3.1.5 Chain

A chained transaction or sequence of transactions are data exchanges which are known before hand and preprogrammed such that prefacing poll request sequences are not required. Multiblock messages may be transferred most efficiently on the bus by establishing a chained linkage via a poll request sequence and then transferring all subsequent message blocks via the chained linkage transfer mechanism. In this way, poll requests are not necessary between every transfer, thereby increasing the effective bus data bandwidth. Unchained message sequences are single or multiblock data transfers which require polling messages between each data transfer. Unchained transactions require a polling message to preface each data transfer.

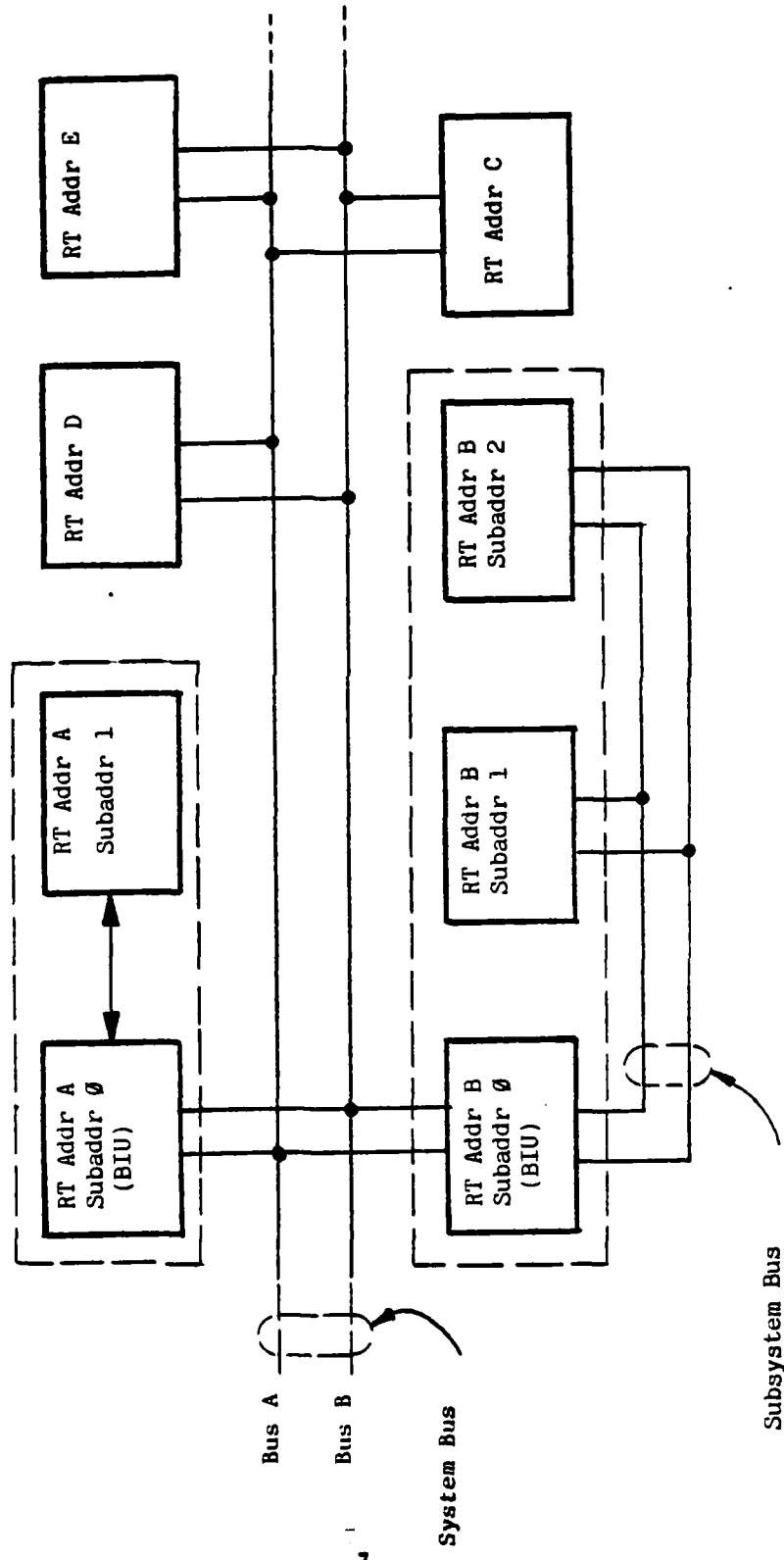
### 3.1.6 Remote Terminal (RT) Address

The RT address is a field consisting of the most significant five bits of the command word and status word. It identifies the hardware destination of the message whether it be a device, subsystem, or class of subsystems. RT address 1F hex (31 decimal) is reserved as the global broadcast address.

### 3.1.7 Subaddress

The subaddress field of the command word and status word consists of the next most significant five bits after the RT address. This field may be used to identify subsystems of a remote terminal or bus controller, internal registers, processes, or functions as required by the system designer. The RT address field and the subaddress field may be rather freely used by the system integrator. For instance, it is within the scope of this protocol that more than one physical unit on the bus (i.e., more than one "RT") may use the same RT address and that the subaddress field will actually select the specific subsystem, process, subfunction, etc. Figures 3.1.7-1 and 3.1.7-2 presents a system level picture of this implementation. Certain advantages can be realized with this scheme in the broadcast mode of operation as presented in sections 3.3.4 and 3.4. For any RT address except the broadcast address subaddress 0 is reserved for the interface controller of the bus unit itself (BC or RT). This functional unit or module is called the Bus Interface Unit (BIU) and is described in section 3.1.26. When RT address 31 (the broadcast address) is specified, the subaddress field takes on the meaning of class or priority number instead of the normal subaddress interpretation. The class or priority number indicates a particular subset of bus units which are permitted to respond or actually accept data. Section 3.3.4 and its subsections defines this mode of operation in more detail. Throughout this document and strictly within the context of the protocol description, the generic term "bus unit" is used to indicate any RT address or subaddress functional unit which may receive and respond to a message (request, suggest or Mode Discrete) or initiate a request. A bus unit will more often be the commonly known RT but it may refer to a subaddress as well.

NADC-81049-50



NOTE: The subaddresses shown are arbitrary.

FIGURE 3.1.7-1 Multiple Bus Unit Implementation of RT Addresses and Subaddresses

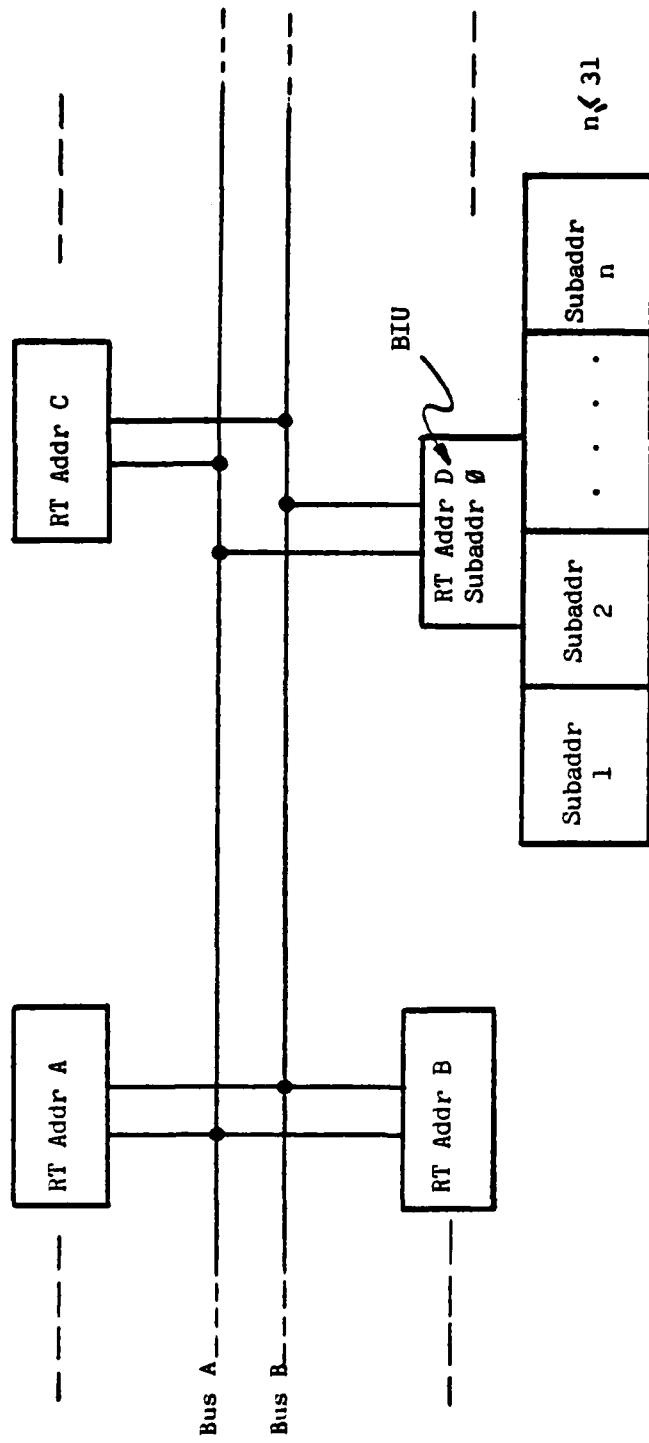


FIGURE 3.1.7-2 Alternative Multiple Bus Unit Implementation of RT Addresses and Subaddresses

### 3.1.8 Command Word (CW)

The command word is the sixteen bit word which is transmitted first when a message is initiated. The contents of the command word are presented in detail in section 3.2.2.1. The synchronization waveform for all word types is defined in section 4.5.

### 3.1.9 Status Word (SW)

The status word is the sixteen bit word which is transmitted first in all message responses. The contents of the SW are presented in detail in section 3.2.3.1.

### 3.1.10 Data Word (DW)

Data words (DW's) are 32-bit words which are found only in data blocks. With one exception, data words are exclusively application dependent and are further broken down into three DW types. These are Normal Data Words (section 3.1.12), Control Data Words (section 3.1.11) and Interrupt Data Words (section 3.1.13). Section 3.2.2.2 (Extended Command Amplifier) describes how these words shall be used to prioritize data types for application use. The exception mentioned above with respect to the use of DW's concerns certain Mode Discrete commands which involve data transfers for protocol control use. In these message types only Normal Data Words (NDW's) shall be used.

### 3.1.11 Control Data Word (CDW)

Control Data Words (CDW's) are 32-bit data words which are involved in Type 1 and Type 3 data transfers (see section 3.2.2.1 on Command Word format). The content of the CDW is completely application program dependent and under this protocol scheme the CDW is not used for any bus interface or protocol control function. Along with the Interrupt Data Word (IDW, subsection 3.1.13) and Normal Data Word (NDW, next subsection) the CDW is intended for use by the system integrator as one mechanism for prioritizing data and message transfers. Section 3.2 details this further.

### 3.1.12 Normal Data Word (NDW)

Normal Data Words (NDW's) are 32-bit data words which are always involved in Type 0 data transfers and may or may not be involved in all of the other types of data transfers (see section 3.2.2.2 on CW+1). The NDW is also involved in Mode Discrete messages which involve data transfers.

### 3.1.13 Interrupt Data Word (IDW)

Interrupt Data Words (IDW's are 32-bit data words which are involved in Type 2 and Type 3 transfers (see section 3.2.2.1 on the Command Word format).

The content of the IDW is completely application program dependent and, as with the CDW, is never used for bus interface or protocol control. It is intended for important or prefacing information that requires immediate attention. As with the CDW and NDW the IDW is included as one mechanism which the system integrator has at his disposal for prioritizing data and message transfers.

### 3.1.14 Mode Discrete

Mode Discretes are commands which are executed and monitored directly by the bus (RT) interface controller hardware. They are defined by the Command Amplifier field of the CW when the Command Code is binary 00 (see section 3.2.2.1 on the Command Word format). It is within the context of this protocol that a Mode Discrete command, although executed and monitored by the bus interface hardware, may be directed towards any subaddress. Therefore, the subaddress field indicates the actual recipient of the Mode Discrete command particularly in the case of subaddress 0 (the bus interface controller subaddress). See section 3.1.26 on the BIU. Certain Mode Discrete commands involve data block transfers.

### 3.1.15 Nesting

Nesting is the mechanism by which one or more distinct transactions may be interleaved on a particular subaddress under BC control. In essence, the protocol allows the implementation of a Last-In-First-Out (LIFO) transaction stack with respect to the subaddress of a bus unit. This function is implemented via bit #'s 5 and 4 of the Extended Command Amplifier (ECA) and the Extended Status Amplifier (ESA). These fields are further detailed in sections 3.2.2.2.2 and 3.2.3.2.2 respectively. Normal communications take place with the Nest field set to 0, the lowest level. Nest = 3 is the highest level. The highest level nested transaction must be completed before the next lower level transaction can be completed. If, during a multiblock transaction (Nest = 0) it is necessary to initiate another message sequence to or from any other bus unit then the Nest field for the new transaction is set to 1. The new transaction, whether single or multiblock, must be completed before the Nest = 0 transaction is continued.

### 3.1.16 Suggest

The suggest is the prefacing command for an actual information transfer. When a suggest is used there is the implicit supposition that the transfer type is expected or known and is ready to take place between two linked bus units. A suggest may be set up by a previous request command (next subsection) or it may be used between bus units where there is only one type of transfer which can take place at that time. See section 3.4 for more detailed information.

### 3.1.17 Request

A request is used to formally initiate a specific message transfer type between two bus units. The current bus controller (section 3.1.22) may use a request or a suggest to initiate a message transfer type. A remote terminal (section 3.1.23) in response to a poll, may only use a request to initiate a message transfer. The bus controller must then come back to the remote terminal with a suggest in order to actually effect the transfer itself. Section 3.4 (Control and Information Exchange Sequences) discusses this area in further detail.

### 3.1.18 Poll Request Sequence (PRS)

A Poll Request Sequence is used to implement the basic poll/response mode of operation of the bus protocol and is one of the four fundamental transaction types. A PRS is always initiated by the current bus controller and consists of a specific mode discrete command to which the receiving remote terminal responds with either a request (either transmit or receive) including status or just status. Figure 3.4.2-1 and section 3.4 describe this message sequence in more detail. The PRS is used in both the normal mode of communications and also in the broadcast mode.

### 3.1.19 Mode Discrete Sequence (MDS)

Mode Discrete Sequence is used to implement or directly modify protocol operation and is one of the four fundamental transaction types. A MDS is always initiated by the current bus controller and consists of a mode discrete command (see section 3.2 of Word Formats and figures 3.2.2.1-1) to a remote terminal followed by a response from the remote terminal consisting of a status frame (section 3.1.25) and up to 1024 NDW's. The content of the response message including the number of NDW's is completely dependent on the particular mode discrete. The MDS is used in the normal mode of communications and also in the broadcast mode (section 3.3.4).

### 3.1.20 Information Transfer Request (ITR)

The Information Transfer Request (ITR) Sequence is used to determine if a particular bus unit can accept or transmit a specific type of information. It is one of the four fundamental transaction types and is only initiated by the current bus controller. Furthermore, the ITR may only be used in the normal mode of communications. The broadcast mode is illegal for the ITR.

### 3.1.21 Information Transfer Suggest (ITS)

The Information Transfer Suggest (ITS) sequence is used to effect the actual information transfer between two bus units. This information transfer is for a particular sequence or data type that has been previously initiated

by an ITR or for a particular sequence or data type that has been preprogrammed between the two bus units. The ITS is one of the four fundamental transaction types and is only initiated by the bus controller. It may be used both in normal communications and in the broadcast mode. Section 3.4.4 details the use of the ITS sequence further.

### 3.1.22 Bus Controller (BC)

A Bus Controller is the bus unit which initiates and controls all transactions on the bus. At any one time there can be only one bus controller although this function may be transferred to another bus unit under direct control of the current bus controller. See also section 3.3.1 for further details.

### 3.1.23 Remote Terminal (RT)

A remote terminal (RT) is any bus unit which is not currently the bus controller. At the discretion of the system integrator any RT may become a bus controller. Section 3.3.2 elaborates RT functions.

### 3.1.24 Command Frame

The command frame is comprised of a CW plus from two to four ECW's and is used to convey all protocol control information from the BC to one or more RT's. The CW and all ECW's in the command frame are sixteen bits long. The content of the command frame is detailed in section 3.2.1.

### 3.1.25 Status Frame

The status frame is comprised of a SW plus from two to four ESW's and is used to convey all protocol response information from the RT to the BC. The SW and all ESW's in the status frame are sixteen bits long. The content of the status frame is detailed in section 3.2.3.

### 3.1.26 Bus Interface Unit (BIU)

The Bus Interface Unit (BIU) is the functional unit consisting of all hardware and firmware necessary to interface a RT to the multiplex data bus. Functionally the BIU implements all hardware and software functions associated with the interface defined in this document - that is, the protocol defined herein. The BIU performs all encoding and decoding necessary to effect the command frames and status frames in all modes of communications defined in sections 3.2, 3.3 and 3.4. Functionally, the BIU shall not perform any application activities with respect to the Data Words described in section 3.1.10.

3.1.27 Extended Command Word (ECW)

Extended Command Words (ECW's) are 16 bit words which follow the command word in the command frame and contain the source information, block and sequence type information and all other information necessary to complete the specification of the desired transaction. There are from two to four ECW's in a command frame depending on the transaction which is specified in the command word. The sync waveform for the ECW is specified in section 4.5 and the definition of the contents of the ECW's is found in section 3.2.2.1. ECW's do not contain any application function dependent information.

3.1.28 Extended Status Word (ESW)

Extended Status Words (ESW's) are 16 bit words which follow the status word in the status frame and contain the remainder of the response information or request information (for a poll) that is not in the status word. Depending on the transaction, the status frame contains from two to four ESW's each with the sync waveform specified in section 4.5. ESW's do not contain any application function dependent information. The subsections under 3.2.3.1 contain detailed definitions of the ESW's.

3.1.29 Bus Control Data

Within the context of this protocol, Bus Control Data refers to the data base resident in the BC and each RT functional unit which contains the parameters necessary to effect communications over the bus. In addition, the data base for each bus unit contains the current state of that bus unit with respect to ongoing bus communications. The bus control data base for a bus unit may vary from system to system but the values that could be expected to be found in it include the following:

- a) Currently active transaction type
- b) Current nest level
- c) Bus units involved in all nested transactions
- d) Requests pending
- e) RT's permitted to transmit data to or receive data from this bus unit
- f) Class/priority number (broadcast)

The BC shall have a copy of the bus control data base for each bus unit available to it. The bus control data base shall only contain data which is required for executing the bus protocol and no application oriented data. The format of the control data base is (TBD).

### 3.1.30 Check Word

In the context of this document, check words are words whose contents reflect an algorithm which has operated on the preceding block of words. Proper interpretation of the check word will indicate the absence or presence of one or more bit errors over the previous data. A check word shall be included as the last ECW of all Command Frames, as the last ESW of all Status Frames, and as an appending data word (DW) to all data blocks. Check words found in Command Frames and Status Frames shall be called Frame Check Words (FCW's) and check words appending data blocks shall be called Data Check Words (DCW's). DCW's are not included as part of the block and word count found in the Block and Residual Word Count ECW and ESW, but are actually part of the protocol control strategy. Check words have unique word type identifiers associated with them, and these can be found in section 4.5. Further information on the usage of check words is found in section 3.3.5.

## 3.2 Word Formats

### 3.2.1 General

Messages are generally broken up into an initiating information transfer and a status response information transfer. The initiating information transfer is made up of a Command Frame plus a possible data block while the status response information is made up of a Status Frame plus a possible data block. The Command Frame is itself made up of a 16 bit CW plus two or more (less than or equal to 4) 16 bit ECW's (i.e., the Command Frame consists of three or more 16 bit words). The data block which may accompany the Command Frame is made up of one or more (less than or equal to 1024) 32 bit DW's plus an appending DCW (not part of the word count). The Status Frame is made up of a 16 bit SW plus one or more (up to 3) 16 bit ESW's (i.e., the Status Frame is made up of two or more 16 bit words). The data block which may accompany the Status Frame is made up of one or more (up to 1024) 32 bit DW's plus an appending DCW (not part of the word count). The following sections and subsections under 3.2 detail the word formats necessary to effect all legal transaction types under this protocol.

### 3.2.1.1 Bit Priority

The most significant bit shall be transmitted first in the data word with the less significant bits following. For the purposes of this protocol, bit #31 is defined as the most significant data word bit. Where multiple precision quantities are to be transmitted (quantities requiring more than 32 bits) the most significant word shall be transmitted first followed by the less significant words in descending order of value. Also for the purposes of this protocol, the most significant bit of the command frame words and status frame words (bit #15) shall be transmitted first with the less significant bits following in descending order by bit number.

### 3.2.1.2 Word Types

The basic word types defined in this protocol document fall into two categories, depending on their function. Those words involved exclusively in protocol implementation and control are all sixteen bits long and are found in the Command Frame and Status Frame. Detailed descriptions of these words will follow in this section. Those words which are involved exclusively in application functions are all thirty-two bits long and are found only in the data blocks. There are three basic types of words which may appear in the data block:

- a) Normal Data Words (NDW's) - described in section 3.1.12.
- b) Control Data Words (CDW's) - described in section 3.1.11.
- c) Interrupt Data Words (IDW's) - described in section 3.1.13.

### 3.2.2 Command Frame

The command frame conveys all protocol information from the BC to one or more RT's and consists of one CW plus two or more ECW's. The following sections and subsections through 3.2.2.1 define the contents and usage of these words. Figure 3.2.2-1 gives the control frame with the words it may contain. Note that the content of the command frame varies with the particular transaction which is active.

#### 3.2.2.1 Command Word (CW)

The detailed CW field definition is shown in figure 3.2.2.1-1. The Command Code (CC) and Command Amplifier (CA) specify the overall protocol action to be taken while the succeeding ECW's contain the parameters necessary to complete the detailed transaction definition. The CC field (bit #'s 5 and 4) specifies the highest level action to be taken with the CA field (bit #'s 3 - 0) providing the next level of detail. Note that the CA field interpretation is dependent on the value of the CC field thereby extending the protocol function definition capability while minimizing the protocol overhead bandwidth. The Command Code specifies three major protocol functions for which each has its own Command Amplifier definition. One Command Code (CC=3) is unassigned and is therefore illegal.

##### 3.2.2.1.1 Mode Discrete Command Code (CC = 0)

A Mode Discrete is an interface protocol command transmitted by the BC to an RT via a Mode Discrete Sequence (MDS). It is intended for execution by and under the control of the RT's BIU (subaddress 0); however, at the discretion of the systems integrator, certain mode discrete commands may be directed to subaddresses other than subaddress 0. Regardless of the subaddress, responsibility for execution, control and status reporting of the Mode Discrete remains with the BIU. The Mode Discrete commands are as follows.

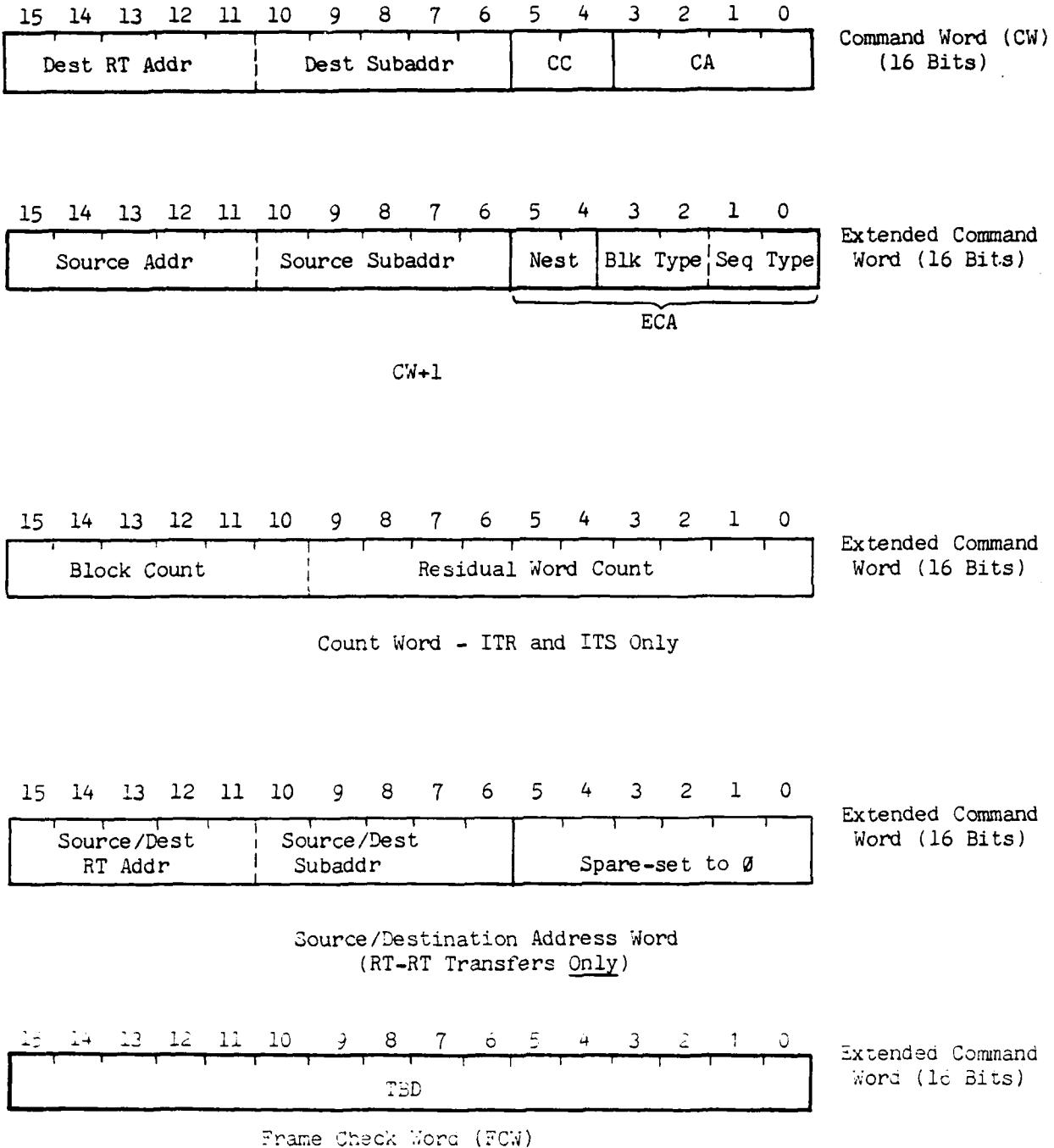


FIGURE 3.2.2-1 Command Frame Content

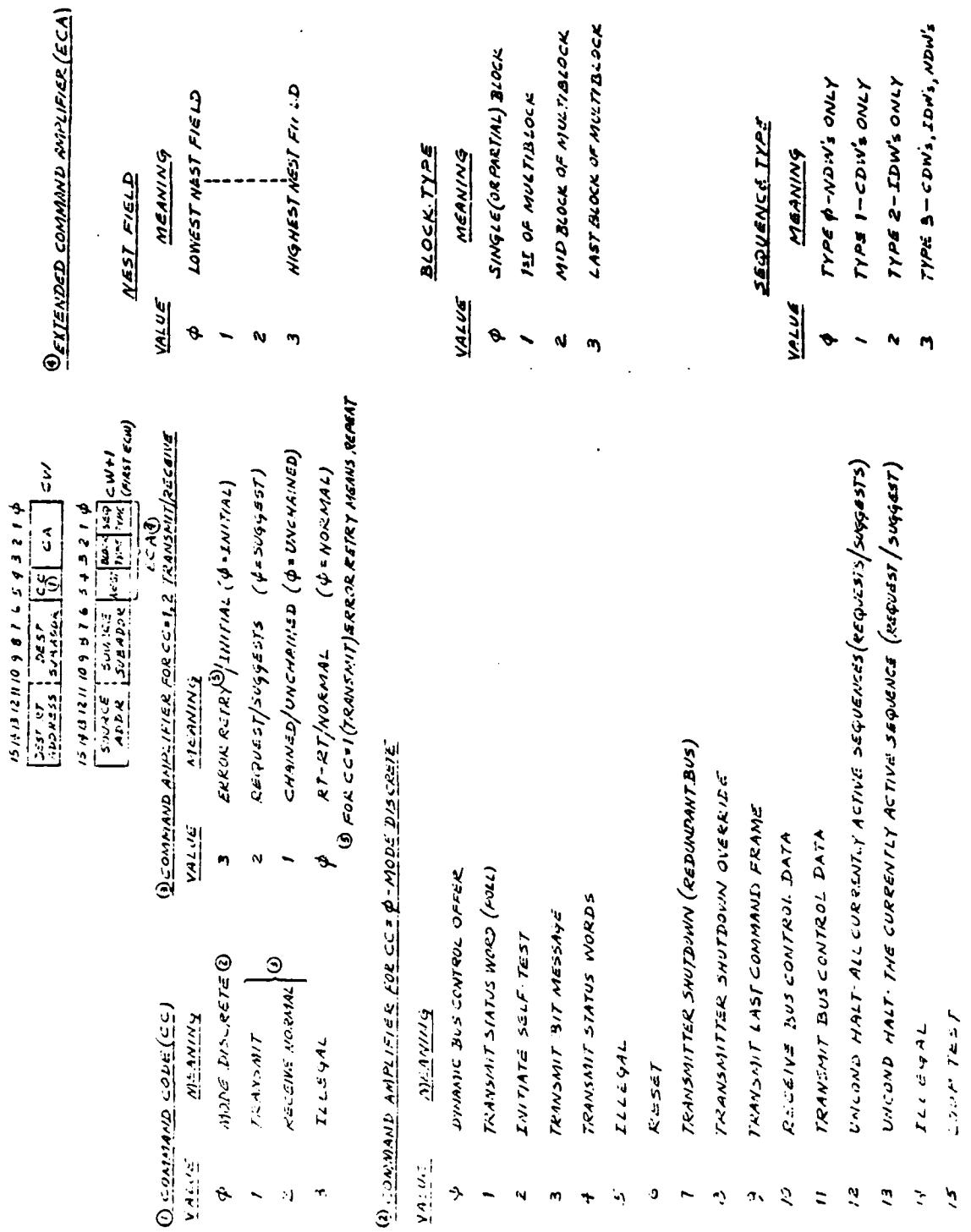


FIGURE 3.2.21-1 COMMAND WORD AND CWT DEFINITION

#### 3.2.2.1.1.1 Dynamic Bus Control Offer (CA = 0)

This mode discrete is an offer by the current BC to transfer total control of the multiplex data bus to a receiving RT and may be used in the normal BC to single RT mode as well as the broadcast mode (section 3.3.4.5). It is normally intended that this command be directed to an RT's BIU (subaddress 0); however, it is within the context of this protocol to use a subaddress other than 0. One example of this situation was given in section 3.1.7 (Subaddress definition).

#### 3.2.2.1.1.2 Transmit Status Word (CA = 1)

This command is used by the BC to both read the status of an RT and to poll an RT or subaddress for service request information. When the command is directed to an RT's BIU (subaddress 0) the response may be generated with respect to subaddress 0 (the subaddress of the BIU) or with respect to another subaddress. In this case, the RT is acting as arbiter with respect to the subsystems or processes (subaddresses) which it is interfacing to the multiplex data bus. This command is valid in both the normal BC to single RT mode and in the broadcast mode (presented in section 3.3.4)

#### 3.2.2.1.1.3 Initiate Self Test (CA = 2)

This command causes the RT to initiate and execute a BIT sequence at the end of which a BIT status word is prepared. As with the previous Mode Discrete commands, Initiate Self Test may be directed to a subaddress other than 0. Initiate Self Test, when directed to subaddress 0 (the BIU), will initiate self test on all RT functions to include all RT subaddress interface functions. Initiate Self Test is illegal in the broadcast mode.

#### 3.2.2.1.1.4 Transmit BIT Message (CA = 3)

This command causes the BIT information associated with the subaddress in the command word to be transmitted as data (DW's) to the BC. A Transmit BIT Message mode discrete to a particular subaddress which is not immediately preceded by an Initiate Self Test command to the same subaddress is an illegal sequence and shall cause an Illegal Sequence error response. It is illegal to use this mode discrete in the broadcast mode.

#### 3.2.2.1.1.5 Transmit Status Words (CA = 4)

This mode discrete command is executed under the same conditions and with the same results as the Transmit Status Word mode discrete (CA = 1) with the additional result that any status words related to the particular subaddress in the CW (bit #'s 9-5) shall also be transmitted as data (DW's) to the BC. Of course in this situation, the Block and Residual Word Count ECW

(section 3.2.2.3) shall be used to indicate the number of words in the data block. The number of DW's associated with a response to this mode discrete shall be restricted to one full data block (up to 1024) and a violation of this (e.g., a block count greater than 1) shall be detected as an error by the BC with the subsequent initiation of the error recovery sequence (section 3.3.5.2). Unlike the Transmit Status Word mode discrete (CA = 1), this mode discrete is illegal in the broadcast mode.

#### 3.2.2.1.1.6 Reset (CA = 6)

This command causes all functions which are associated with the subaddress in the command word to be reset. A reset to subaddress 0 will cause all RT functions to be cleared. This command is illegal in the broadcast mode.

#### 3.2.1.1.7 Transmitter Shutdown (CA = 7)

This command will cause the RT to immediately disable the transmitter on the redundant bus. It is illegal in the broadcast mode.

#### 3.2.2.1.1.8 Transmitter Shutdown Override (CA = 8)

This command causes the RT to enable (or re-enable) the transmitter on the redundant multiplex data bus. It is illegal in the broadcast mode.

#### 3.2.2.1.1.9 Transmit Last Command Frame (CA = 9)

This mode discrete causes the BIU to transmit as data (DW's) the last command frame from the subaddress indicated by the current command word. When addressed to subaddress 0, this command will cause the last command frame received by the RT to be transmitted regardless of which subaddress had received it. This command is illegal in the broadcast mode.

#### 3.2.2.1.1.10 Receive Bus Control Data (CA = 10)

This mode discrete command is accompanied by data in one or more data blocks which contain parameters necessary for the RT with its subaddresses to carry on communications over the multiplex data bus. These parameters include, on an as needed basis, the unique RT subaddress time out values for broadcast operation, the RT subaddress class numbers (also used for broadcast operation), (TBD). Figure (TBD) shows the Command Frame and the general format for the Bus Control Data Block(s). This mode discrete represents one of the three exceptions mentioned previously wherein the data words are used for a more application oriented purpose. Bus control data involved with this command is never application oriented but only contains protocol control parameters.

3.2.2.1.1.11 Transmit Bus Control Data (CA = 11)

This mode discrete is used to transfer Bus Control Data from a RT to a BC or to another RT and as such it complements the Receive Bus Control Data mode discrete. As with the Receive Bus Control Data mode discrete this command is an exception in that the data words are used for a non application oriented purpose. This command is also illegal in the broadcast mode.

3.2.2.1.1.12 Unconditional Halt - All Currently Active Sequences (CA = 12)

When an RT receives this mode discrete all currently active suggests and requests for the intended subaddress are immediately terminated. If the subaddress in the command word is 0 then all suggests and requests for all subaddresses are terminated. With respect to the Nesting function presented in sections 3.2.2.2 and 3.4.5, this mode discrete in essence "pops" the entire sequence stack, i.e., reinitializes the sequence stack pointer with respect to the destination subaddress. This command is treated as a NOOP if no sequence is active. The broadcast command is illegal for this mode discrete.

3.2.2.1.1.13 Unconditional Halt - The Currently Active Sequence (CA = 13)

This mode discrete causes the currently active request or suggest to be immediately terminated. In essence the Nest "stack" is popped once by this command for the intended subaddress. As with the immediately preceding command, if the subaddress is 0 then the currently active sequence is terminated for all subaddresses. Furthermore, if no sequence is active then this mode discrete is treated as a NOOP. This command is illegal in the broadcast mode.

3.2.2.1.1.14 Loop Test (CA = 15)

The Loop Test command initializes a preprogrammed sequence of transactions between the BC and RT. Immediately following the transaction containing the Loop Test mode discrete the BC will transmit messages with predefined data patterns which the RT will echo back. Section 3.4.6 defines the exact loop test sequence. It is illegal to use this mode discrete in the broadcast mode.

3.2.2.1.2 Transmit Command Code (CC = 1)

The Transmit function provides the mechanism for transferring data from an RT to either the BC or to another RT depending on the content of the Command Amplifier (CC) field. Reference figure 3.2.2.1-1 for the Transmit word format. The CA field for the Transmit function contains discrete bits which may be set or cleared depending on the required transmit functions. The Transmit function is illegal in the broadcast mode. Bit definitions for the CA field are presented in the following four subsections.

#### 3.2.2.1.2.1 Error Retry/Initial (Bit #3)

This bit, when set to a logic 1, indicates that the RT is to repeat the previous message. The subsequent ECW's in the Command Frame must be identical to the original Transmit Command Frame. Also, the other bits in this Command Amplifier must be identical to the original CA field in the original transmit CW. When this bit is cleared to a logic 0, it indicates that this is an original Transmit Command Frame.

#### 3.2.2.1.2.2 Request/Suggest (Bit #2)

This bit, when set to a logic 1, indicates that a transmit request for a particular data and sequence type is active. The RT is required to respond as to whether or not the requested information is available. The possible RT responses are defined in sections 3.2.3 and its subsections. When this bit is cleared to a logic 0 it indicates that the BC expects or is ready to receive the data described by the other parameters in the Command Frame and that the data should accompany the RT response.

#### 3.2.2.1.2.3 Chained/Unchained (Bit #1)

This bit is set to a logic 1 when the current transaction or sequence of transactions meet the requirements of section 3.1.5 for chained sequences.

#### 3.2.2.1.2.4 RT-RT/Normal (Bit #0)

This bit, when set to a logic 1, indicates that the BC is requesting/suggesting that the RT transmit a message to another RT. The succeeding ECW's in this Command Frame contain all additional relevant information concerning the transaction(s), including the target RT address and subaddress (see section 3.3.3). When this bit is cleared to a logic 0 the BC is requesting/suggesting that the RT transmit the specified message back to the BC. This bit explicitly specifies when the RT-RT transfer is to take place.

#### 3.2.2.1.3 Receive Command Code (CC = 2)

The Receive function provides the mechanism for transferring data from either the BC or an RT to another RT and as such it complements the Transmit function and is symmetric to it. The Command Amplifier field for the Receive function, as with the Transmit function, is comprised of discrete bits which may be set or cleared depending on the required receive function. The symmetry of this function with the Transmit function comes about as a result of the CA field bits having similar meanings to those of the Transmit CA field. The Receive function is legal in both the normal modes of communication (BC-RT or RT-RT) and also in the broadcast mode. Bit definitions for the CA field are presented in the following four subsections.

### 3.2.2.1.3.1 Error Retry/Initial (Bit #3)

This bit, when set to a logic 1, indicates that the intended message for the RT to receive is a repeat of the previous message. The succeeding ECW's in the Command Frame must be identical to the ECW's in the original Command Frame as must the other bits of the CA field. If this bit is cleared to a logic 0 then the intended message is an original transmission to the RT.

### 3.2.2.1.3.2 Request/Suggest (Bit #2)

This bit, when set to a logic 1, indicates that a receive request for a particular data and sequence type is active. The RT is required to respond as to whether or not it is ready to receive the indicated message. The permissible RT responses are defined in section 3.2.3 and its subsections. When this bit is cleared to a logic 0, it indicates that the message is accompanying the Command Frame and that the BC expects the RT to accept it.

### 3.2.2.1.3.3 Chained/Unchained (Bit #1)

This bit is set to a logic 1 when the current transaction or sequence of transactions meet the requirements of section 3.1.5 for chained sequences.

### 3.2.2.1.3.4 RT-RT/Normal (Bit #0)

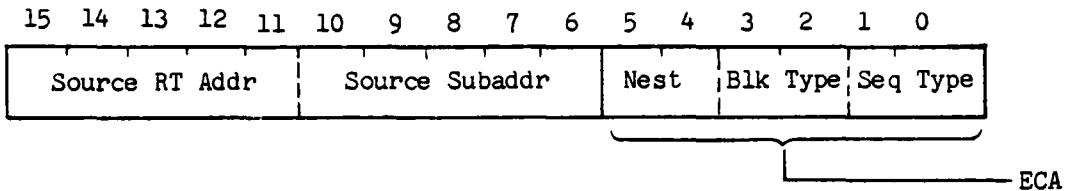
This bit, when set to a logic 1, indicates that the BC is requesting/suggesting that the RT receive a message from another RT. The succeeding ECW's in this Command Frame contain all additional relevant information concerning the transaction including the source RT address and subaddress (see section 3.3.3). When this bit is cleared to a logic 0, the BC is requesting/suggesting that the RT receive the specified message from the BC. This bit explicitly specifies when an RT-RT transfer is to take place.

## 3.2.2.2 CW+1

CW+1 is the 16 bit ECW immediately following the CW and appears in all Command Frames for all transactions. This word contains the source information of the message plus the Extended Command Amplifier (ECA) field. The ECA field is further broken up into three 2-bit fields which further detail the type of message sequence, block indicator information and whether or not the message is nested. Figure 3.2.2.2-1 shows the format of the CW+1 ECW and the bit field allocation.

### 3.2.2.2.1 Source Information Field (Bit #'s 15-16)

This field is broken up into two subfields which contain the source RT address and subaddress of the message. The format and bit location of this



Nest Level (Bit #'s 5, 4)

<u>Value</u>	<u>Meaning</u>	
0	First or Lowest Level	Highest level nested sequence must be completed first before lower level nest can be completed.
3	Last or Highest Level	

Block Type (Bit #'s 3,2)

<u>Value</u>	<u>Meaning</u>
0	Single/Partial Block
1	First of Multi Block
2	Mid Block of Multi Block
3	Last Block of Multi Block

Sequence Type (Bit #'s 1, 0)

<u>Value</u>	<u>Meaning</u>
0	Type 0 - NDW's Only
1	Type 1 - CDW's, NDW's
2	Type 2 - IDW's, NDW's
3	Type 3 - CDW's, IDW's, NDW's

FIGURE 3.2.2.2-1 CW+1 - Source Information and Extended Command Amplifier (ECA)

field are exactly the same as in the Command Word. This field provides a dual capability for the RT designer. It allows an enhanced integrity test to determine if the bus unit which is trying to communicate is actually permitted to access the RT plus it provides the destination information necessary for the status response.

### 3.2.2.2.2 Extended Comand Amplifier (ECA, Bit #'s 5-0)

The ECA field provides supplementary information concerning the current transaction including block type, sequence type and nest level. This field is irrelevant (set to logic 0) for Mode Discrete commands which do not involve data transfers but is required for all other messages.

#### 3.2.2.2.2.1 Nest Field (Bit #'s 5, 4)

The Nest Field indicates the nest level of the request/suggest as defined in section 3.4.5. It is illegal to initiate or attempt to execute a transaction specifying an inappropriate nest level. Should this occur, an Illegal Sequence Error will result.

#### 3.2.2.2.2.2 Block Type (Bit #'s 3, 2)

The Block Type indicates generally whether the current message is a single (or partial) block transfer or whether it is part of a multi block transfer. If the current message is part of a multi block transfer this field indicates whether it is the first block of a multi block transfer, whether it is a mid-block of a multi block transfer or whether it is the last block of a multi block transfer. For a message to be a mid-block of a multi block transfer, there must be at least three blocks involved in the multi block transfer. A two block transfer has a First Block and a Last Block but no Mid-block. The Last Block code is used to signal the end of a multi block transfer thereby precluding the requirement for a subsequent poll for more data to or from the RT (unless it is necessary to initiate a subsequent linkage). Depending on the total number of words to be transferred it is possible that the Last Block of a multi block transfer may not be a complete 1024 word block but only a partial block.

#### 3.2.2.2.2.3 Sequence Type (Bit #'s 1, 0)

This field specifies the make-up of the current data block in terms of NDW's, CDW's and IDW's. The table below further defines the data block content:

<u>Sequence Type</u>	<u>Meaning - Data Block Content</u>
0	at least 1 NDW

- 1 at least 1 CDW and up to 1023 NDW's
- 2 at least 1 IDW and up to 1023 NDW's
- 3 at least 1 CDW, at least 1 IDW and up to 1022 NDW's

The actual definition of the number of each DW Type in the data block is left to the System Integrator; however, the use of each DW Type shall be as specified in sections 3.1.11, 3.1.12 and 3.1.13.

### 3.2.2.3 Block and Residual Word Count

The bit field allocation for this ECW is shown in figure 3.2.2-1 and the content is the binary representation of the total number of full data blocks to be transferred and the binary representation of the residual word count in the last block (up to 1023). For multi block transfers the block count is decremented for each successive data block transfer until, for the last block to be transferred, the block count will be 0 and the residual word count field will reflect the number of words in the last, partial block. This word is only present in Command and Status Frames for which data block transfers are involved and shall not appear otherwise. Violation of this will cause an Illegal Sequence Error to be detected.

### 3.2.2.4 Source/Destination Word

The Source/Destination ECW is only included in the Command Frame when an RT-RT transfer is active. The format for this word is shown in figure 3.2.2-1. The Source/Destination RT address and subaddress appear exactly as they would appear in the CW or SW with the remainder of the word being zero filled. For non RT-RT transfers, this ECW will not be included in the Command or Status Frames. Violation of this will cause an Illegal Sequence Error.

### 3.2.2.5 Frame Check ECW

A Frame Check Word (FCW) shall be included as the last ECW in all Command Frames, as shown in figure 3.2.2-1. This FCW shall be generated by the BC protocol processing function and verified by the RT protocol processing function. This ECW shall neither be passed to nor originate from any host application program. Section 3.3.5 contains implementation and error processing information for the FCW.

### 3.2.3 Status Frame

The status frame provides all protocol response information from the RT to either the BC or to another RT (RT-RT transfers) and consists of one SW plus two or more ESW's. The following sections and subsections through 3.2.3.5

define the contents and usage of these words and their relationships with the words which make up the command frame. Figure 3.2.3-1 gives the status frame with the words that it may contain. Note that the content of the status frame varies with the particular transaction which is active.

### 3.2.3.1 Status Word (SW)

The detailed SW bit field definition is shown in figure 3.2.3.1-1. As can be seen, the format of the status word is to a large extent symmetric with that of the command word in that there is a two bit Status Code (SC) field and a four bit Status Amplifier (SA) field. These fields contain the highest level protocol response information with the following ESW's providing the more detailed response information. As with the command word, the SA field definition is dependent on the value of the SC field. The following four subsections present the detailed definitions of the SW fields.

#### 3.2.3.1.1 Idle/MDS Response (SC = 0)

This status code is used to transfer five basic types of information to the BC. These types are as follows:

- a) All sequences terminated; idle state.
- b) Busy status (BIU or subsystem/subaddress)
- c) Mode discrete responses
- d) Sequence cancellation
- e) Bus control acceptance

In keeping with the philosophy of this protocol, it is appropriate that a SW response be applicable to a subaddress contained in the command word as in the sense described in section 3.1.14 (Subaddress definition). For data transfer responses, it is implicit that the appropriate subaddress involved be reflected in the SW subaddress field. For the most part the SA field for Idle/MDS Response responds to the Mode Discrete Command Amplifier field (for CC = 0) on a one to one basis; however, there are minor deviations and for this reason the SA field definitions for SC = 0 shown in figure 3.2.3.1-1 are defined in more detail in the following subsections.

##### 3.2.3.1.1.1 Bus Control Acceptance (SA = 0)

This is the response which is generated upon acceptance of a valid Dynamic Bus Control Offer.

NADC-81049-50

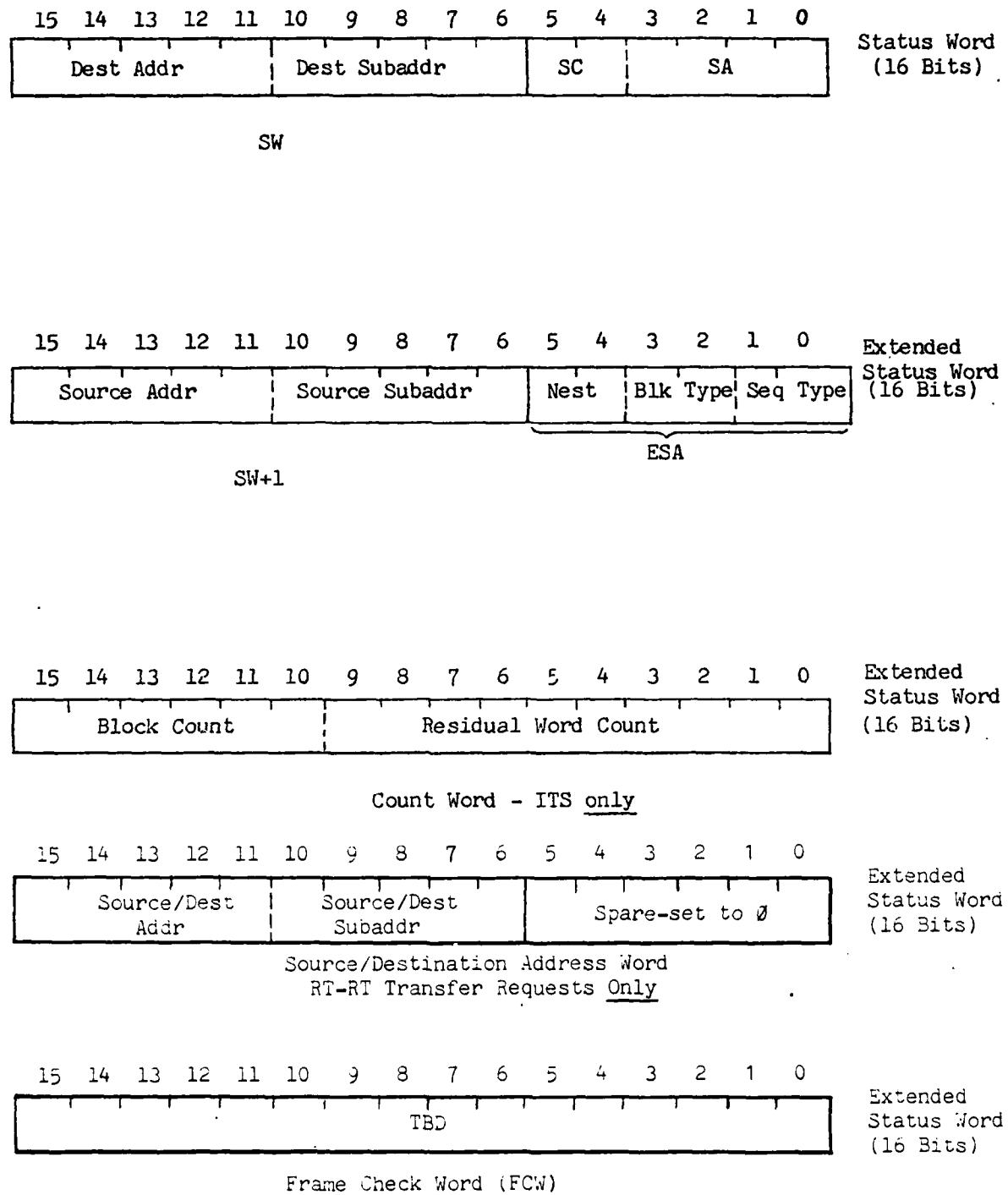


FIGURE 3.2.3-1 Status Frame Content

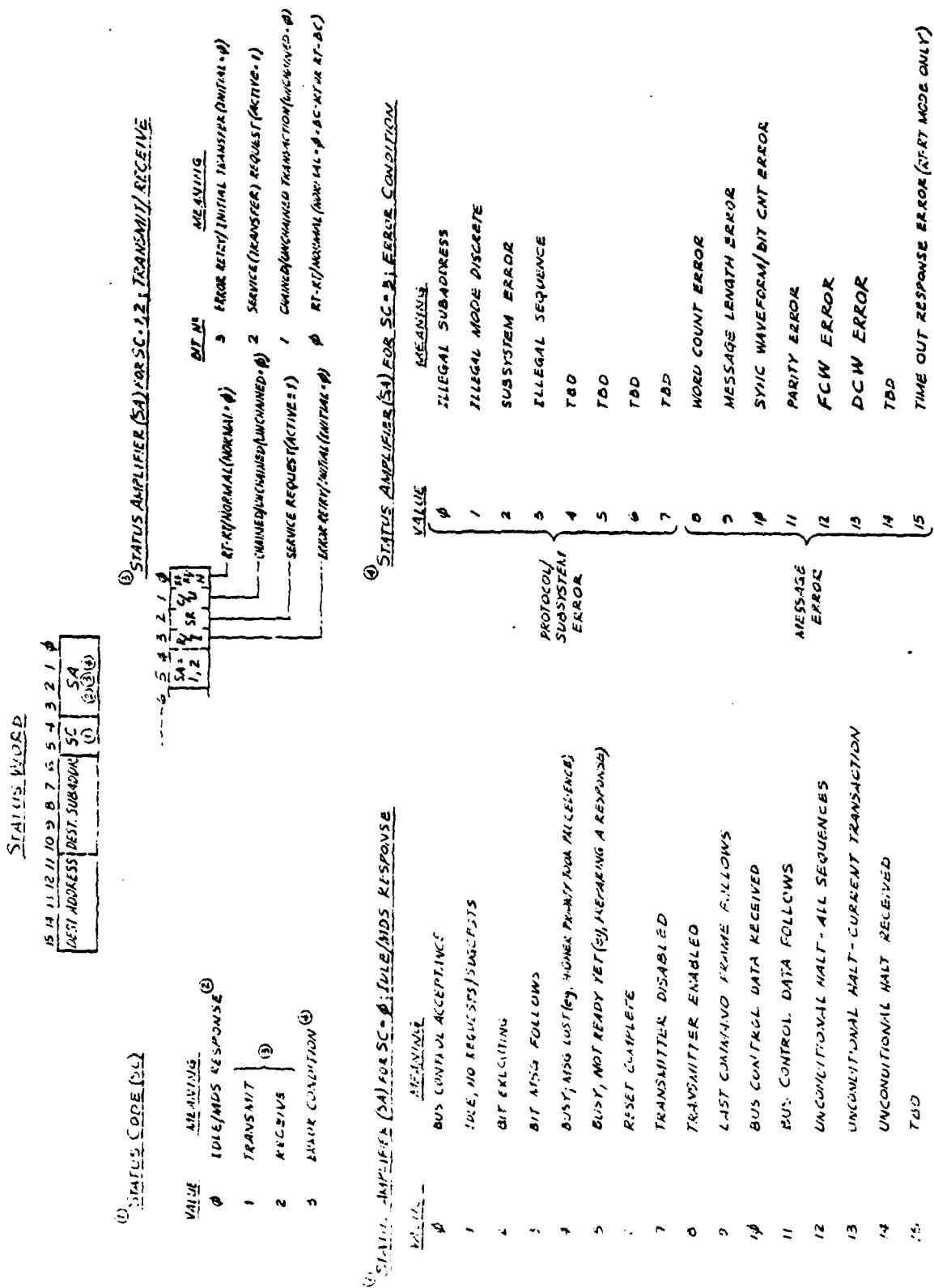


FIGURE 3.2:3.1-1 Status Word Definition

3.2.3.1.1.2 Idle; No Requests/Suggests (SA = 1)

This response is generated as a result of either a valid poll or a valid status request and one or both of the following two conditions is also true:

- a) There are no active transactions.
- b) There are no Requests pending.

3.2.3.1.1.3 BIT Executing (SA = 2)

Upon the valid receipt of the Initiate Self-test Mode Discrete (CC = 0, CA = 2) this status response is generated. Upon completion of the self-test execution function the status is returned to Idle (SA = 1). The BIT Executing status is the only legal response to a valid poll until the BIT function has terminated except in the case of a Reset Mode Discrete which supercedes all functions and commands.

3.2.3.1.1.4 BIT Message Follows (SA = 3)

This status is generated in response to a valid Transmit BIT Message Mode Discrete (CA = 3). The BIT information is included in the accompanying DW block as part of this status message. The content of the BIT message is target application system dependent and is defined at system design time.

3.2.3.1.1.5 Busy; Message Lost (SA = 4)

This status response indicates that the addressee was busy and was unable to process a valid received message. The requester should reinitiate the transaction at a later time. This response is not a legal response to a Reset Mode Discrete (CA = 6).

3.2.3.1.1.6 Busy; Not Ready Yet (SA = 5)

This response is generated when the addressee cannot prepare a more appropriate response in the defined basic response time period but has received and is processing the message. This response is illegal for a Reset Mode Discrete (CA = 6).

3.2.3.1.1.7 Reset Complete (SA = 6)

This response is generated upon the receipt of a valid Reset Mode Discrete (CA = 6) and the completion of the reset function. The Reset function shall be completed and the status response generated within the basic time out response period.

3.2.3.1.1.8 Transmitter Disabled (SA = 7)

This response is generated upon the valid receipt of the Transmitter Shutdown Mode Discrete and after the transmitter for the redundant bus has been disabled. These functions shall be completed within the basic response time out period and, therefore, a Busy response shall be illegal.

3.2.3.1.1.9 Transmitter Enabled (SA = 8)

Similar to the Transmitter Disabled response this response is generated upon the valid receipt of a Transmitter Shutdown Override Mode Discrete (CA = 8) and after the transmitter for the redundant bus has been enabled. These functions shall be completed within the basic response time out period and, therefore, a Busy response shall be illegal.

3.2.3.1.1.10 Last Command Frame Follows (SA = 9)

This response is generated upon the valid receipt of a Transmit Last Command Frame Mode Discrete (CA = 9). The previously received command frame accompanies this status response in the appended DW block.

3.2.3.1.1.11 Bus Control Data Received (SA = 10)

This response is generated upon the successful receipt of the Bus Control Data Block (section 3.1.29).

3.2.3.1.1.12 Bus Control Data Follows (SA = 11)

This response is generated upon receipt of a valid Transmit Bus Control Data Mode Discrete (CA = 11). The global information is included as data in the accompanying DW block.

3.2.3.1.1.13 Unconditional Halt-All Sequences (SA = 12)

This status is generated when it is desired to terminate all currently active suggests and requests by the addressee. This is in the form of a request and no action is taken by the addressee until or unless the BC transmits an Unconditional Halt - All Currently Active Sequences (CA = 12, CC = 0). This function essentially "pops" all nested transactions involving the addressee (i.e., clears the nest stack). This status is illegal in response to a Mode Discrete command except for a Poll (Status) Request. During a multiblock transaction the Service Request bit of the SA field (sections 3.2.3.1.2 and 3.2.3.1.3) may be set, thereby "interrupting" the BC and causing a Poll to be transmitted. The addressee may then respond to the poll with an Unconditional Halt request.

3.2.3.1.1.14 Unconditional Halt - The Currently Active Sequence (SA = 13)

This response is identical to the previous one with the single exception that it applies only to the currently active transaction. If only one transaction is active, then this response is in fact exactly the same as the previous one. Again, the status is illegal in response to all Mode Discrete commands except for the Poll (Status) Request. Invocation of this function may be accomplished in the same manner as described in the previous subsection (section 3.2.3.1.1.13).

3.2.3.1.1.15 Unconditional Halt Received (SA = 14)

This status is generated upon the valid receipt of either of the Unconditional Halt Mode Discretes (CA = 12, 13) and completes the unconditional halt sequence of messages. At this time the appropriate transactions are terminated.

3.2.3.1.2 Transmit (SC = 1)

This status code shall be used to indicate the availability of data to be transmitted in the following two situations:

- a) As a response to a poll or status request (CC = 0, CA = 1) this status code together with the Service Request bit (bit #2) shall indicate that data is available for transmission.
- b) During actual data transmissions as a result of receiving a transmit command code this status code shall be used as a positive acknowledgement that there is data to follow in the message.

This status code has its own unique bit interpretation of the SA field as seen in figure 3.2.3.1-1. These definitions are presented as follows.

3.2.3.1.2.1 Error Retry/Initial (Bit #3)

This bit, when set to a logic 1, indicates that the current message being transmitted is a repeat of the previous message. This bit shall only be set in response to the Error Retry bit being set in the Command Frame (CC = 1, Bit #3 = 1).

This bit being cleared to a logic 0 indicates that the current message is the first to be transmitted.

3.2.3.1.2.2 Service (Transaction) Request (Bit #2)

This bit, when set to a logic 1, indicates that a transaction request is pending. The Service Request is the mechanism by which the BC is initially

informed of RT initiated nested requests during active multiblock transactions. Section 3.4.2 (Poll Requests), section 3.4.3 (Information Transfer Requests) and section 3.4.5 (Nesting) further describe the use of this function.

#### 3.2.3.1.2.3 Chained/Unchained (Bit #1)

This bit, when set to a logic 1, indicates that the referenced transaction is chained according to the definition in section 3.1.5.

#### 3.2.3.1.2.4 RT-RT/Normal (Bit 0)

This bit, when set to a logic 1, indicates that the referenced transaction is an RT-RT transaction. See section 3.3.3 (RT-RT Mode) for further details on usage. When this bit is at a logic 0 the referenced transaction is taking place between the BC and the addressee (RT or RT subsystem).

#### 3.2.3.1.3 Receive (SC = 2)

This status code, like the Transmit status code, has a unique SA field interpretation that is similar in structure to that of the Transmit status code. The Receive status code shall be used to reflect two distinct response conditions depending on whether the response is to a poll (status) request or to a message involving some form of data.

- a) The Receive status code when used as a response to a poll (status) request shall be used together with the Service Request bit of the SA field to indicate that a Receive data request condition is pending.
- b) The Receive status code when used as a response to a receive data message transfer shall indicate a positive acknowledgement to the received data (assuming no errors were detected). In this situation if the Service Request bit is also set to a logic 1, the interpretation shall be that the Receive status code acknowledges the received data message and also that an additional service request condition is pending (possibly due to an asynchronous or event driven situation such as an interrupt) which requires an additional polling step interleaved (nested) with the active transaction. More detail on the usage of the receive status code is reserved for section 3.4.2 (Poll Requests), section 3.4.3 (Information Transfer Requests) and section 3.4.5 (Nesting). The following subsections define in further detail the SA field interpretation for the Receive status code.

#### 3.2.3.1.3.1 Error Retry/Initial (Bit #3)

This bit, when set to a logic 1, indicates that the received message is a repeat of the previous message and should not be processed as new data. When at a logic 0 state, this bit shall indicate that the received message is new.

3.2.3.1.3.2 Service (Transaction) Request (Bit #2)

This bit, when set to a logic 1, shall indicate that a new service request condition is pending. The context of this occurrence defines what further action is required. Section 3.2.3.1.3 (b) and the sections that are referenced therein, fully specify the usage of this bit. When at a logic 0 state, this bit indicates that no further service request conditions are pending.

3.2.3.1.3.3 Chained/Unchained (Bit #1)

This bit, when set to a logic 1, indicates that the referenced transaction is chained according to definition in section 3.1.5.

3.2.3.1.3.4 RT-RT/Normal (Bit #0)

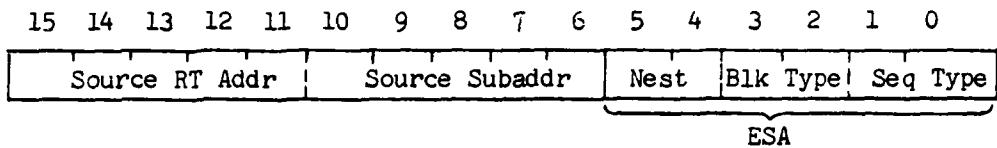
This bit, when set to a logic 1, indicates that the referenced transaction is an RT-RT transaction. See section 3.3.3 (RT-RT Mode) for further details on RT-RT Mode operation. When this bit is at a logic 0, the referenced transaction is taking place between the BC and the addressee (RT or RT subsystem).

3.2.3.1.4 Error Condition (SC = 3)

This status code is generated whenever a message or protocol error is detected. Figure 3.2.3.1-1 shows the breakdown between message errors and protocol errors and their SA field assignments for the Error Condition status code. Further detailed descriptions of these errors and of the responsibility for error recovery and error recovery modes is referred to in section 3.3.5.

3.2.3.2 SW + 1

SW + 1 is the 16 bit ESW immediately following the SW and appears in all Status Frames for all transactions. This word contains the source information of the message plus the Extended Status Amplifier (ESA) field. The ESA field is further broken up into three 2-bit fields which further detail the status response as to the type of message sequence, block indicator information and whether or not the message is nested. SW + 1 shall be implemented as a positive acknowledgement of CW + 1 and therefore it reflects the format of CW + 1 exactly. Figure 3.2.3.2-1 shows the format of SW + 1 and its bit field allocation.



Nest Level (Bit #'s 5, 4) - Response (Ack) or Request

Value    Meaning

0	First or Lowest Level	}	—	Highest level nested sequence must be completed first before lower level nest can be completed.
3	Last or Highest Level			

Block Type (Bit #'s 3, 2) - Response (Ack) or Request

Value    Meaning

0	Single/Partial Block
1	First of Multi Block
2	Mid Block of Multi Block
3	Last Block of Multi Block

Sequence Type (Bit #'s 1, 0) - Response (Ack) or Request

Value    Meaning

0	Type 0 - NDW's only
1	Type 1 - CDW's, NDW's
2	Type 2 - IDW's, NDW's
3	Type 3 - CDW's, IDW's, NDW's

FIGURE 3.2.3.2-1 SW+1 - Source Information and Extended Status Amplifier (ESA)

3.2.3.2.1 Source Information Field (Bit #'s 15-6)

This field is defined exactly as the Source Information Field of CW + 1 except that the contents are the RT address and subaddress of the responding bus unit. The contents of this field should be the same as the contents of the RT address and subaddress fields of the received CW.

3.2.3.2.2 Extended Status Amplifier (ESA, Bit #'s 5-0)

The ESA field provides supplementary information concerning the current acknowledgement or requested message including block type, sequence type and nest level. This field is irrelevant (set to logic 0) for Mode Discrete commands which do not involve data transfers but is required for all other messages.

3.2.3.2.2.1 Nest Field (Bit #'s 5,4)

The Nest Field of SW + 1 indicates the level of the request currently active in the addressed unit or is a positive acknowledgement of the current message nest level. The context of its usage (see section 3.4) shall determine the appropriate interpretation. It is illegal to request or execute a transaction specifying an inappropriate nest level. Should this occur, an Illegal Sequence Error will result.

3.2.3.2.2.2 Block Type (Bit #'s 3,2)

The Block Type indicates whether the currently requested message/transaction is a single (or partial) block transfer or whether it is part of a multi block transfer. Alternatively, the Block Type is a positive acknowledgement of the Block Type in CW + 1, thereby providing redundant error detection capability. The context of usage determines which interpretation is appropriate (see section 3.4). Other than these differences, the Block Type in SW + 1 has the same meaning as its counterpart in CW + 1 (section 3.2.2.2.2).

3.2.3.2.2.3 Sequence Type (Bit #'s, 1, 0)

This field shall be used to indicate either the sequence type of the requested message or to positively acknowledge the current message sequence

type depending on the context usage. The detailed meaning of this field is the same as that of its counter-part in CW + 1 (see section 3.2.2.2.2.3). For clarity the table is repeated here:

<u>Sequence Type</u>	<u>Meaning - Data Block Content</u>
0	at least 1 NDW
1	at least 1 CDW and up to 1023 NDW's
2	at least 1 IDW and up to 1023 NDW's
3	at least 1 CDW, at least 1 IDW and up to 1022 NDW's

### 3.2.3.3 Block and Residual Word Count ESW

The Block and Residual Word Count ESW is used in the dual role as the block and word count specification in a transaction request and as a positive acknowledgement for the current count in a data transfer transaction. Again, the context of usage determines the appropriate interpretation. This ESW is only present in data transfer requests and in data transfer acknowledgement status frames. Violation will result in an Illegal Sequence Error.

### 3.2.3.4 Source/Destination ESW

The Source/Destination ESW is only present in RT-RT transactions and contains the RT address and subaddress of the bus unit which is to transmit or receive the referenced message(s). An Illegal Sequence Error will result if this ESW is included in other than RT-RT transaction situations. See section 3.3.3 (RT-RT Mode).

### 3.2.3.5 Frame Check ESW

A Frame Check Word (FCW) shall be included as the last ESW in all Status Frames as shown in figure 3.2.3-1. This FCW shall be generated by the RT protocol processing function and verified by the BC protocol processing function. This ESW shall neither be passed to nor originate from any host application program. Section 3.3.5 contains implementation and error processing information for the FCW.

## 3.3 Operating Modes

Control of the bus is maintained at all times by a bus unit operating in the bus controller mode. Only one bus unit at a time may be the bus controller; however, provisions are provided within the framework of the protocol to dynamically reallocate this function under the control of the current BC. The following subsections detail all basic operating modes of this protocol.

### 3.3.1 Bus Controller Mode

All control of bus operations is maintained via the bus controller. The BC initiates all Poll (Status) Requests, all Information Transfer Requests and Suggests, all Mode Discrete messages and the transfer of the bus control function to another bus unit. The BC also controls RT-RT transactions and is responsible for monitoring execution of this sequence. The current BC controls the transfer of bus control to another bus unit and does not relinquish control until the handshaking procedure described in section 3.3.4 is successfully completed.

### 3.3.2 Remote Terminal Mode

A remote terminal (RT) may never initiate the transmission of a message on the data bus but must first wait to respond to a PRS, MDS, ITS or ITR (see section 3.4 on Control and Information Exchange Sequences). An RT may initiate a transaction, however, via a request in response to a poll (PRS) or via setting the Service Request bit in the status word if the requirement arises during a multiblock data transfer. An RT may or may not have the capability to assume the role of BC at the discretion of the system designer.

### 3.3.3 RT-RT Mode - General

An RT-RT transaction is implemented under the complete control of the BC and provides a means of transferring either single or multiblock information from one RT to another RT. The actual information transfer is initiated by the BC which transmits two command frames contiguously. The first command frame is an RT-RT RxS to the receiving RT while the second, following in a contiguous fashion, is an RT-RT TxS to the transmitting RT. Within a valid response time out interval, the transmitting RT will transmit the specified message to the receiving RT which will then respond to the BC with status. The sequence of messages which make up an RT-RT transaction is determined by whether the transaction is chained or unchained. These two cases are detailed in the following two subsections and in figures 3.3.3-1 and 3.3.3-2. Under this protocol, the RT-RT capability is not required for implementation except as determined by systems designers for particular applications. This function represents a level of complexity in BC and RT design and implementation which is not strictly necessary for basic intersubsystem data communications and, even though there are overhead and information bandwidth advantages to be gained, it is realized that there are and will be systems that do not require this added performance. Therefore, in order to provide an area of possible cost reduction in system development and maintenance, the RT-RT function is optional under this protocol. For systems which do not utilize this function the RT-RT/Normal bit shall always be set to a logic 0 state. Broadcast RT-RT transactions are legal under this protocol and are defined in section 3.3.3.3. With respect to the method for acknowledgement of broadcast RT-RT transfers, the rules governing normal broadcast data transfers shall apply. These are covered in section 3.3.4.

### 3.3.3.1 Chained RT-RT Transactions

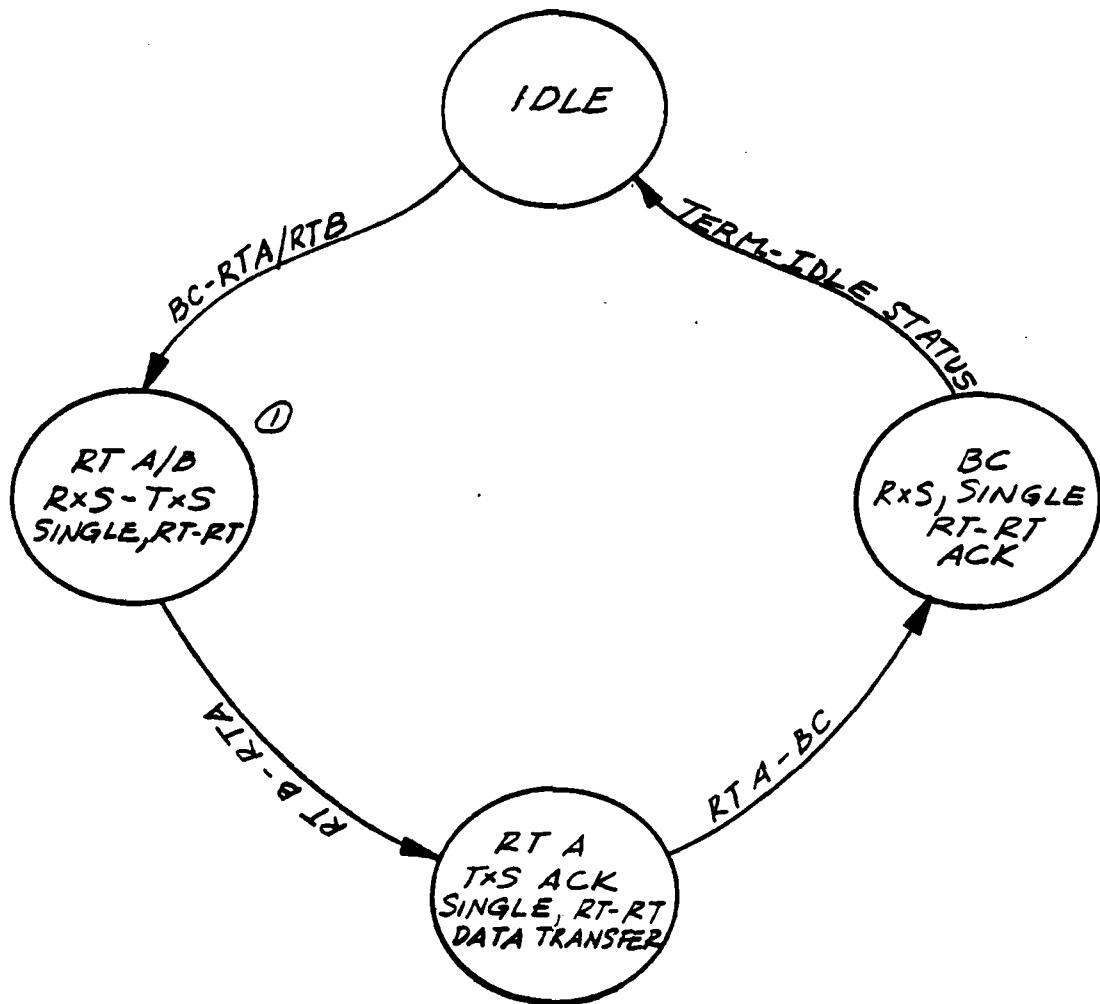
A chained RT-RT transaction may be effectively used when there is a priori knowledge that data is ready to be transferred from one RT to another RT thus precluding the need for prefacing request messages. To execute an RT-RT transaction the BC will transmit an RT-RT Receive Suggest (RxS) command frame addressed to the receiving bus unit followed contiguously by an RT-RT Transmit Suggest (TxS) command frame addressed to the transmitting bus unit. Each command frame shall contain the appropriate message parameters including the respective Source/Destination ECW's and with the RT-RT and Chained/Unchained bits set. Within a valid response time out period, the transmitting bus unit shall transmit a TxS ACK status frame addressed to the receiving bus unit with the appropriate data block following in a contiguous fashion. After receiving the data and assuming that no errors were detected the receiving bus unit shall transmit an RxS ACK status frame to the BC. Should the transmitting bus unit detect an error in the TxS command frame from the BC it shall follow the error recovery steps described in section 3.3.5. Should the receiving bus unit detect any errors during the RT-RT transaction it shall follow the error recovery steps presented in section 3.3.5. Retries and/or any other error recovery measures shall only be initiated by the BC. Figure 3.3.3-1 is representative of a chained RT-RT transaction including the appropriate detailed command and status frame definition.

### 3.3.3.2 Unchained RT-RT Transactions

When the BC has no a priori knowledge of whether a linkage is possible for an RT-RT transaction between two bus units it shall execute an unchained RT-RT transaction. The BC shall issue an RT-RT transmit request (TxR) command frame to the transmitting bus unit in order to determine if it is ready and an RT-RT receive request (RxR) command frame to the receiving bus unit in order to determine if it is ready. The order of execution of the requests is a system design consideration. The request sequence may be executed and repeated on an as needed basis and within the constraints of the target system bus timing cycles until both bus units are ready. The BC shall then initiate an RT-RT Information Transfer Suggest (ITS) as described under Chained RT-RT Transactions in the previous subsection but with the Chained/Unchained bit cleared to a logic 0 state. Figure 3.3.3-2 is representative of an unchained RT-RT transaction. Section 3.4 contains more information on the individual message types which make up RT-RT transactions.

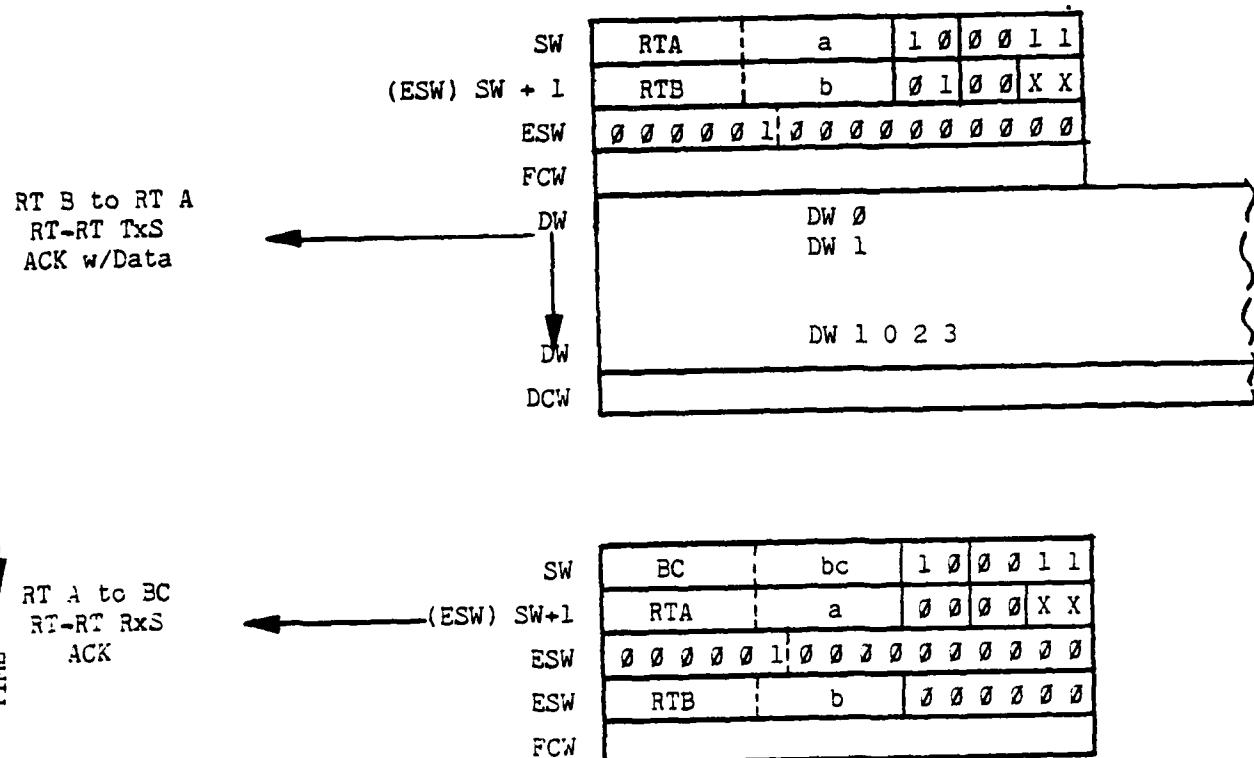
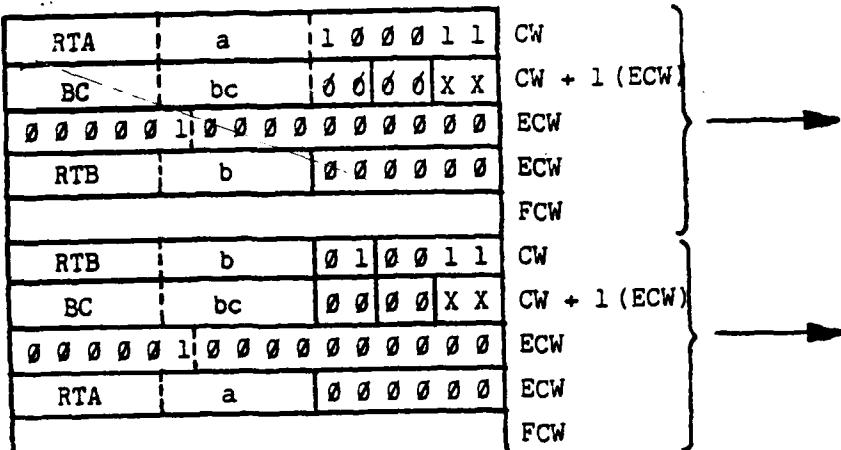
### 3.3.3.3 Broadcast RT-RT Transactions

The broadcast RT-RT transaction provides a mechanism for a bus unit which is not currently the BC to transfer a single data block or multiple data blocks to more than one bus unit simultaneously. The BC controls this transaction in the same way that it controls normal RT-RT transfers along with the added constraints of data transfers under the broadcast mode found in section 3.3.4. The transaction is initiated by the BC transmitting an RT-RT RxS command frame addressed to RT address 31 followed contiguously by an RT-RT TxS command frame addressed to the bus unit which is to broadcast the data. Within a valid response time out interval, the transmitting bus unit shall transmit a TxS ACK status frame addressed to RT address 31 and with the proper



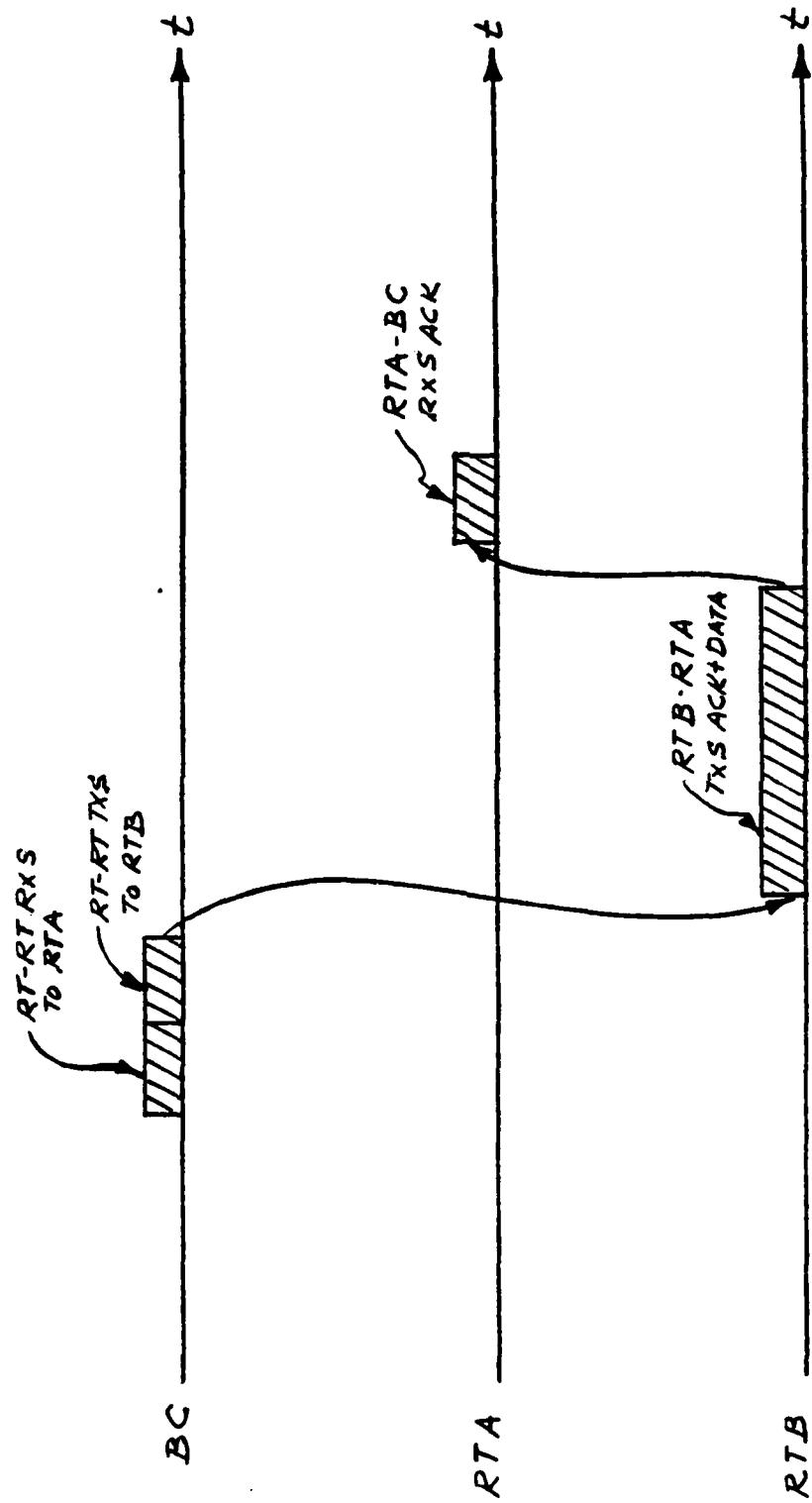
NOTES: 1. See Figure 3.3.3-1(b) for detailed command frame contents.  
 2. Single block RT-RT data transfer shown.  
 3. Service Request, error states not shown.

FIGURE 3.3.3-1(a) Chained RT-RT Transaction High Level State Diagram



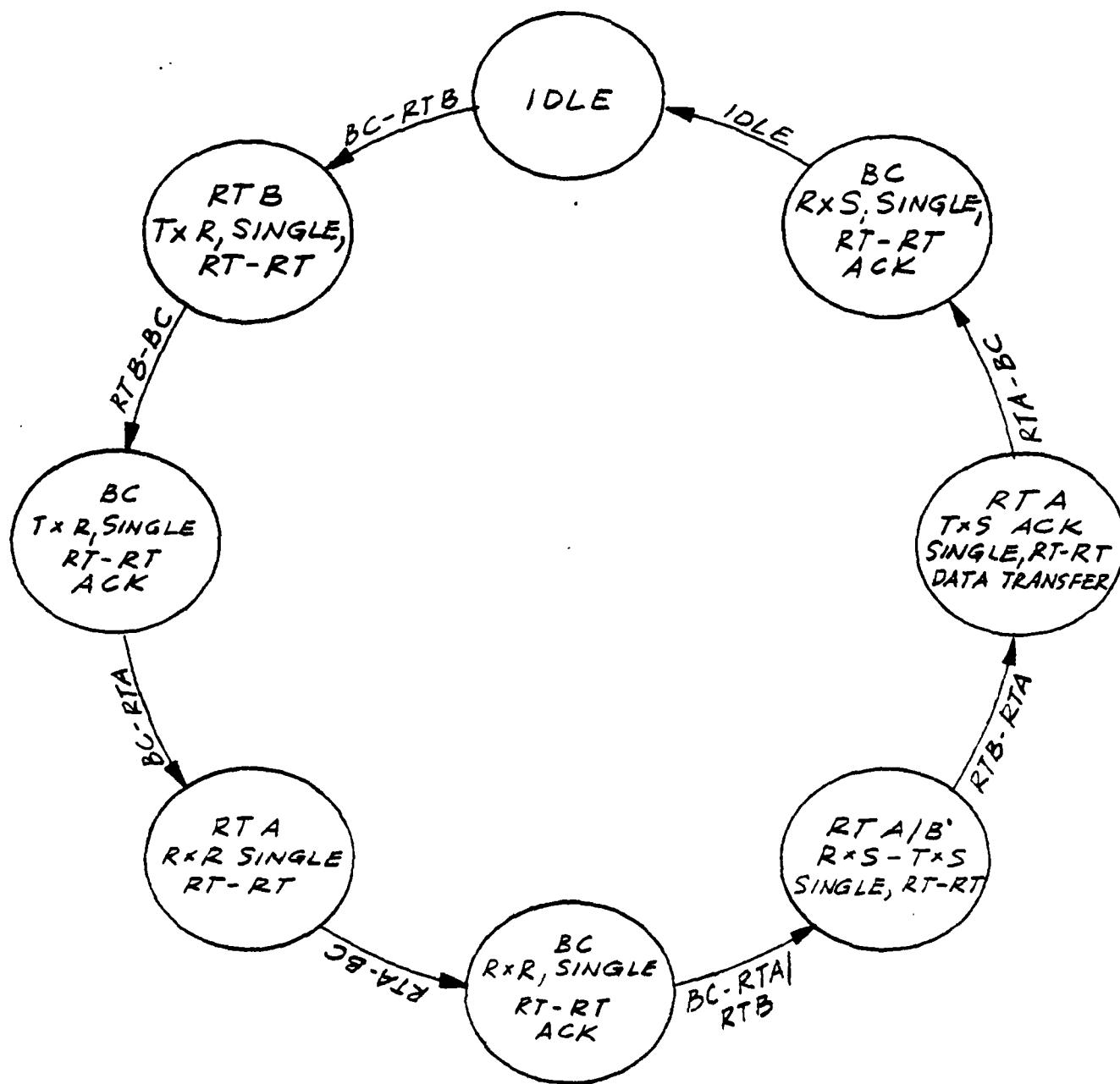
Notes: 1. Single block, 1024 DW RT-RT transfer shown  
2. Sequence type indicated by x's ("don't cares") - Data dependent

FIGURE 3.3.3-1(b) Chained RT-RT Transaction. Detailed Command Frame and Status Frame Content



NOTES: 1. Sequence of messages is the same as in Figures 3.3.3-1(a)-(b).  
 2. Error and Busy messages not shown.

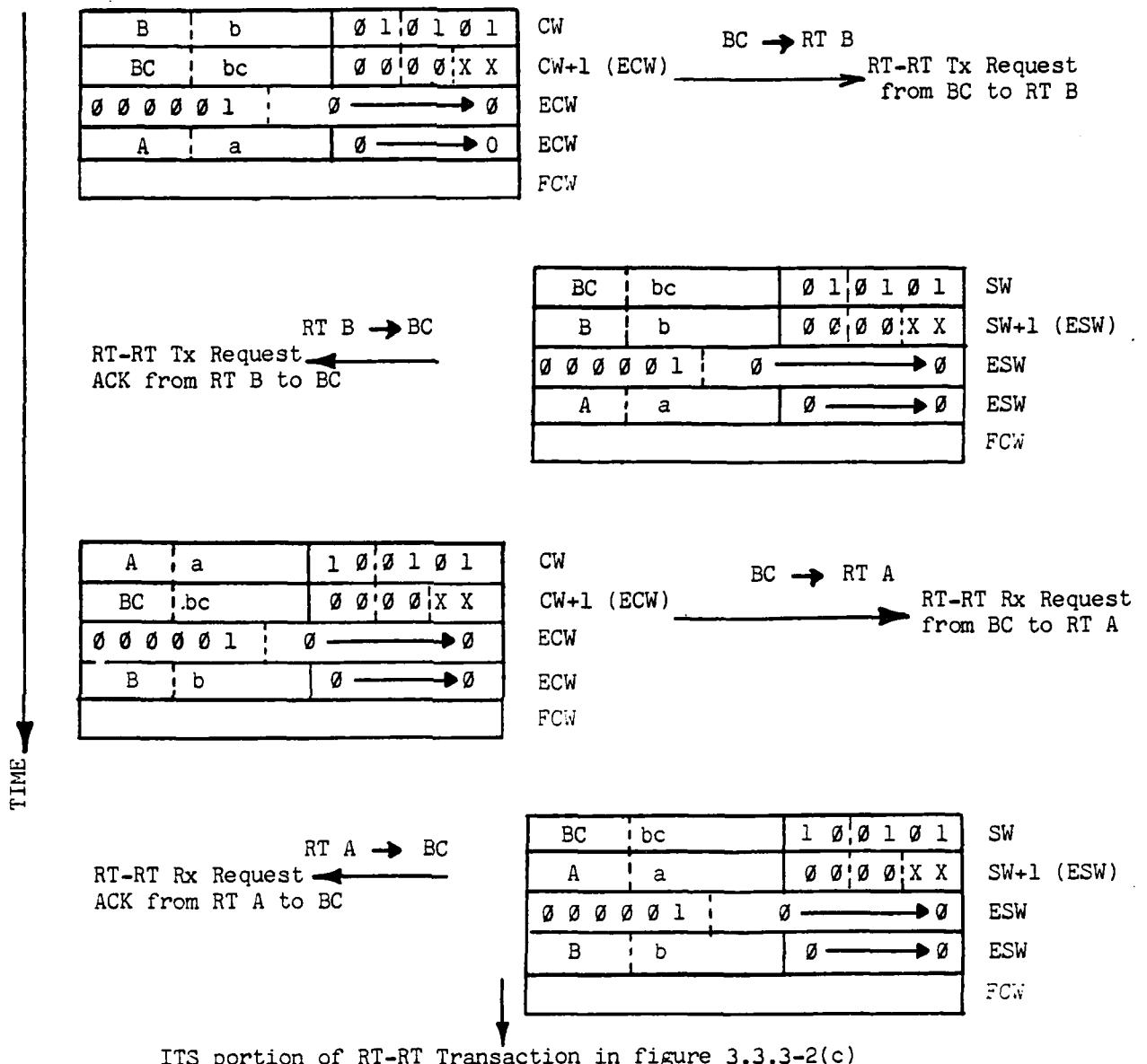
FIGURE 3.3.3-1(c) Chained RT-RT ITS Transaction - Time Line Illustration



NOTES:

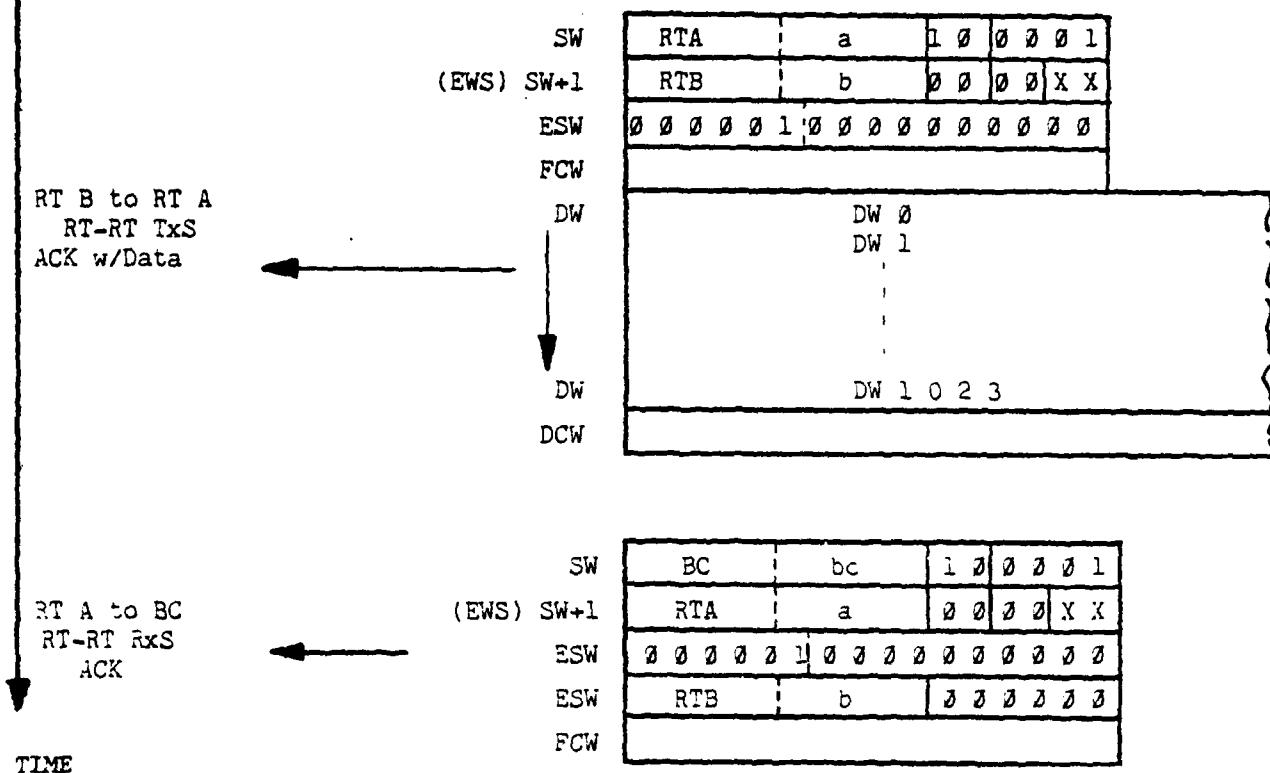
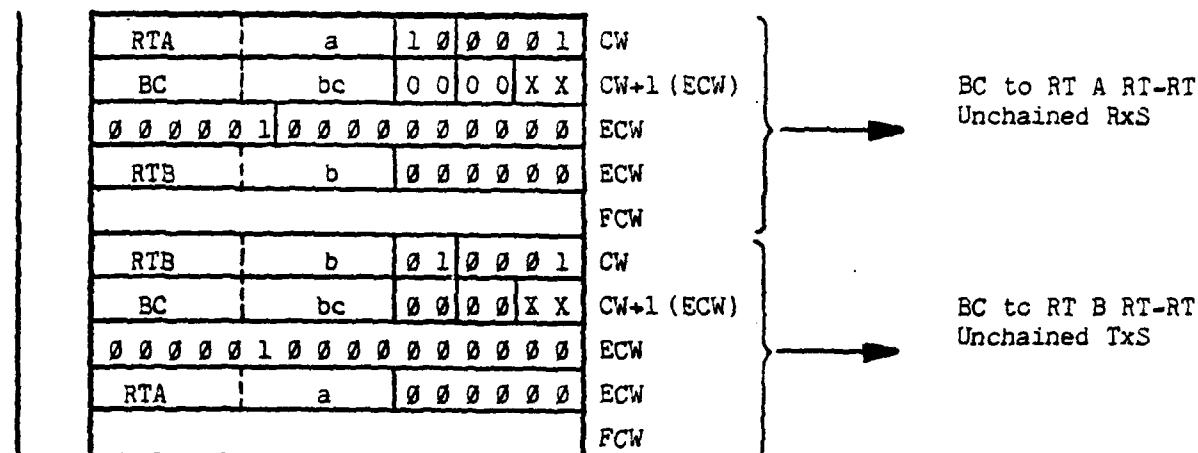
1. See Figures 3.3.3-2(b)-(c) for detailed message content.
2. Single block RT-RT data transfer shown.
3. Service Request, Error, and Busy states not shown.

FIGURE 3.3.3-2(a) Unchained RT-RT Transaction High Level State Diagram



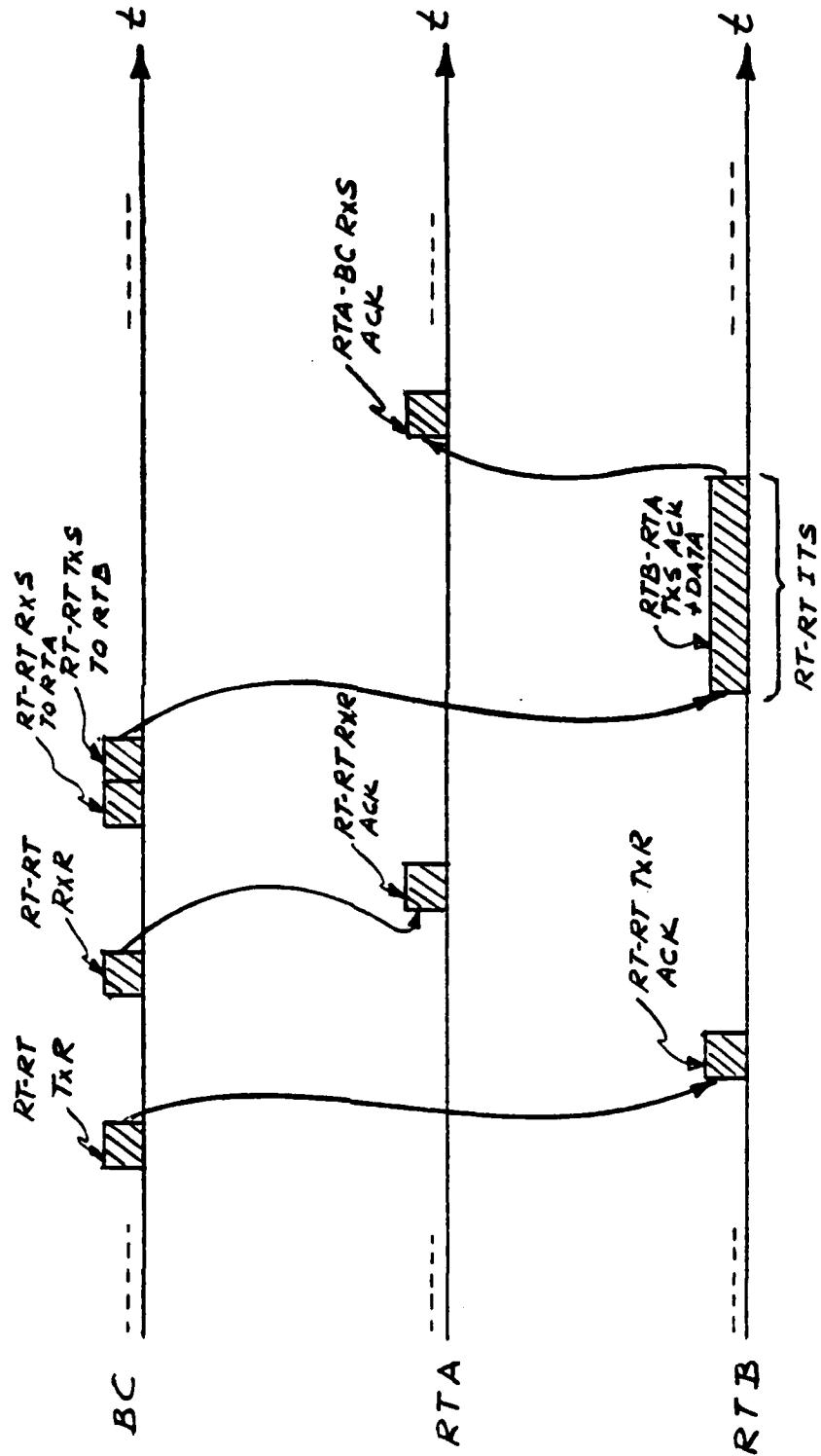
NOTES: 1. Sequence type indicated by x's ("don't cares") - Data dependent  
2. Single block, 1024 DW RT-RT request/acknowledge sequence of unchained RT-RT transaction shown

FIGURE 3.3.3-2(b) Request Messages of Unchained RT-RT Transaction. Detailed Command Frame and Status Frame Content



1. Single block, 1024 DW RT-RT transfer shown
2. Sequence type indicated by x's ("don't cares") - Data dependent

FIGURE 3.3.3-2(c) Unchained RT-RT Message Sequence - Data Transfer Portion



NOTE: Sequence shown is the same as that of Figures 3.3.3-2(a)-(c).

FIGURE 3.3.3-2(d) Single Block Unchained RT-RT ITS Transaction - Time Line Illustration

class/priority value in the subaddress field as specified by the BC in the broadcast RxS. The TxS ACK status frame to address 31 shall be followed in a contiguous fashion by the specified data. Upon receipt of the data by each intended bus unit there shall be no immediate response by any bus unit. Status shall be generated within each receiving unit and stored until the BC polls each of the receivers as specified for broadcast data transfers in section 3.3.4.4. Error detection and reporting shall meet all requirements for both RT-RT and broadcast modes of operation. Operation and execution of broadcast RT-RT transactions are intended to be consistent with target system communication requirements and consistent with the RT-RT and broadcast modes of operation.

### 3.3.4 Broadcast Operation

Broadcast operation as defined herein is a function which allows a BC to communicate with more than one bus unit simultaneously and still maintain a measure of communication control integrity. The broadcast function is implemented via RT address 31 such that a message which is transmitted by the BC to RT address 31 is received by multiple bus units (the exact number of which is determined by the system designer). There are generally two types of messages which may be involved in broadcasts. These are:

- a) Mode Discrete messages which do not involve data transfers.
- b) Any data transfer message (including a Mode Discrete which involves a data block) from the BC to multiple bus units.

It is illegal for a broadcast message to require data transfers to the BC from multiple bus units. Should the BC attempt this transfer there shall be no response by any bus unit and, additionally, the Illegal Sequence error status shall be set in all bus units which are implemented with the broadcast function. The response of bus units to broadcast Mode Discrete commands and the acceptance of data transfers from the BC shall be based on the class or priority assignment found in the subaddress field of the broadcast CW and on the unique time out period designated for each bus unit. These two parameters are detailed in the following two subsections. With respect to non-data broadcast Mode Discrete messages, the first bus unit to respond precludes any other bus unit from responding; therefore, to maximize the advantage of using the broadcast feature it is defined as an Illegal Sequence Error for a bus unit to respond "Busy" to a broadcast Mode Discrete. Likewise, for subsystems which are implemented to accept broadcast data transfers it shall be a system constraint that these subsystems be provided with adequate buffering in order to preclude a "Busy" status from being generated. This protocol supports (on an as needed, system dependent basis) broadcast RT-RT transactions as defined in section 3.3.3.3 and within the context of and according to the rules specified for both RT-RT and broadcast modes of information transfer.

### 3.3.4.1 Time Out (T.O.) Mechanism

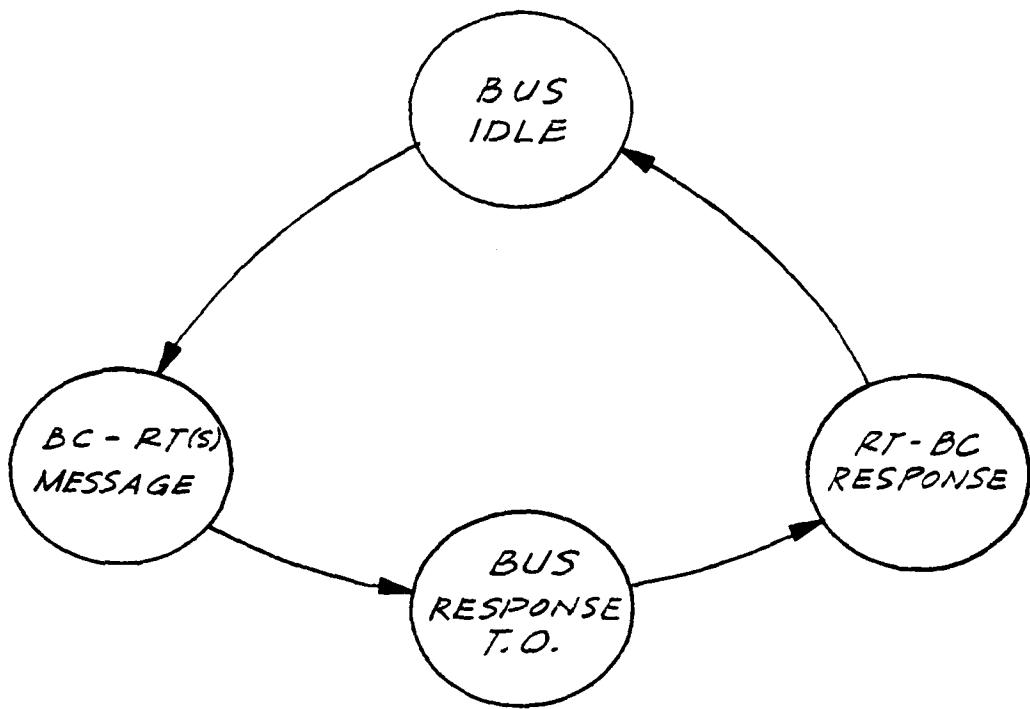
Bus units which are implemented with the broadcast capability shall be assigned a unique response time out period based on their 5 bit class or priority number. No two bus units with the same class or priority number shall have the same response time out value; however, these two parameters shall be dynamically modifiable via the Bus Control Data Base mechanism and they may be modified only by the current bus controller via a Receive Bus Control Data Mode Discrete command. This constraint will ensure that the BC is always updated with the latest bus communication parameters for all bus units. The response time out period following a broadcast Mode Discrete starts at the mid bit transition of the parity bit of the last word in the broadcast message (Manchester signal coding assumed) and includes the basic response gap period ( $T_{BG}$ ) plus integral multiples of the priority response delay time ( $T_{PRD}$ ). These values shall be system dependent; however, they shall also be selectable in 1 usec (microsecond) increments over a range of (TBD). Figures 3.3.4.1-1 and 3.3.4.1-2 illustrate the implementation of  $T_{BG}$  and  $T_{PRD}$ . Section 4.4 details time out response periods further.

### 3.3.4.2 Class/Priority Allocation and the T.O. Mechanism

As mentioned in the previous subsection, each bus unit which is implemented for broadcast operation is assigned a 5 bit class or priority number. This value shall be dynamically modifiable by the BC via Bus Control Data Mode Discrete commands and shall determine whether a bus unit is one of the actual recipients of a broadcast transfer. If the class/priority number in the command frame subaddress field of the broadcast message does not match that of a particular bus unit, then that bus unit shall ignore the message. If the class/priority number does match that of the receiving bus unit, then one of two possible actions will take place.

- a) If the broadcast message involves received data, then the receiving bus unit will not respond but will generate a status response and wait for a status request.
- b) If the broadcast message was a non-data Mode Discrete and the class/priority number matched that of the receiving bus unit, then the bus unit will time out for its unique time out interval and transmit a status frame back to the BC only if no other bus unit responds sooner. Therefore, each bus unit that is enabled to respond by virtue of the received class/priority number must listen in on the bus while it times out. If no other unit responds before a particular bus unit times out, then that unit may respond.

For broadcast messages only, the subaddress field of the CW is used for the class/priority number. Also, a bus unit may be assigned more than one number and each one may have associated with it a different time out period at the discretion of the system designer. Should this field be designated as a priority number (as opposed to an arbitrary class assignment) then a value of 31 (decimal) shall be the highest priority and 0 shall be the lowest.



1. For normal message sequences (BC to single RT) the Bus Response Time Out (T.O.) period is the basic response gap time ( $T_{BG}$ ).

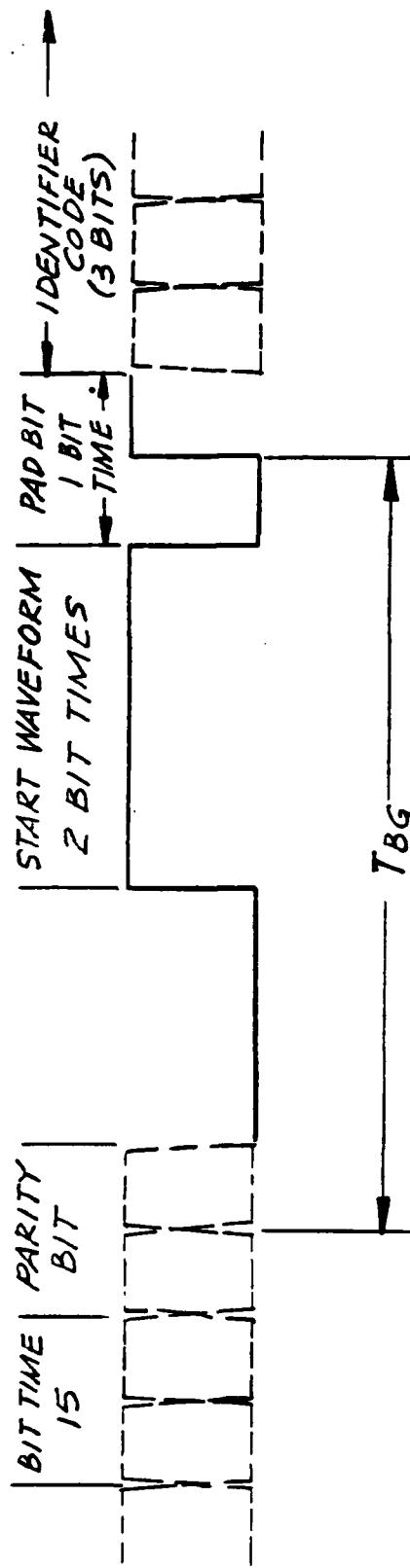
2. For Broadcast Poll Transactions the length of the Bus Response T.O. period will be dependent on the RT with the highest priority service request pending and will be the basic response gap time plus an integral multiple of the priority response delay time.

$$T_R = T_{BG} + \alpha T_{PRD}$$

$\alpha$  = preassigned priority response number (integer)

$$0 \leq \alpha \leq 31$$

FIGURE 3.3.4.1-1 High Level Bus State Transition Diagram; BC-RT Poll Transaction (Normal or Broadcast) or Transaction Request Message Sequence (Normal only)



NOTES:

1. Gap time measurement is from mid bit transition of last parity bit to mid bit transition of Pad Bit.
2. Manchester II coding assumed.
3. See Section 4.5 for details on start waveform and identifier code.
4. Pad Bit is a Manchester coded 0.

FIGURE 3.3.4.1-2 Response Time Out Interval Measurement

### 3.3.4.3 Polling and Broadcast

Under this protocol, the BC may transmit a Transmit Status Word (poll) Mode Discrete in the broadcast mode following the guidelines set forth in sections 3.3.4, 3.3.4.1 and 3.3.4.2. This use of the broadcast function is one method for providing a means of informing the BC of event driven conditions (i.e., interrupt situations in more conventional systems). Figure 3.3.4.3-1 illustrates a typical broadcast poll transaction. If there is no response to this broadcast poll, then the BC may change the class/priority number and repeat the transaction.

### 3.3.4.4 Data Block Transfers and Broadcast

Data block transfers in the broadcast mode may only occur as messages from the BC to multiple bus units or as broadcast RT-RT messages and will follow the conditions set forth in sections 3.3.3.3, 3.3.4, 3.3.4.1 and 3.3.4.2. Typical examples of broadcast data block transactions are the broadcast Receive Bus Control Data Mode Discrete and broadcast Receive Suggest (see section 3.4.4.2 for the general Receive Suggest transaction). Following a broadcast data block transfer there shall not be an automatic status response by any bus unit. Instead, it shall be part of this protocol that the BC shall individually poll each of the intended receiving bus units (intended with respect to the class/priority number) for status of the received broadcast message. The status polling shall take place immediately following the broadcast message and within the constraints of the basic bus timing cycles of the particular system. Error recovery, if necessary, shall proceed as detailed in section 3.3.5. This application of the broadcast mode is designed to help optimize data bus information bandwidth and still maintain message integrity and bus communication control. Figure 3.3.4.4-1 illustrates a broadcast Receive Suggest.

### 3.3.4.5 Dynamic Bus Allocation and Broadcast

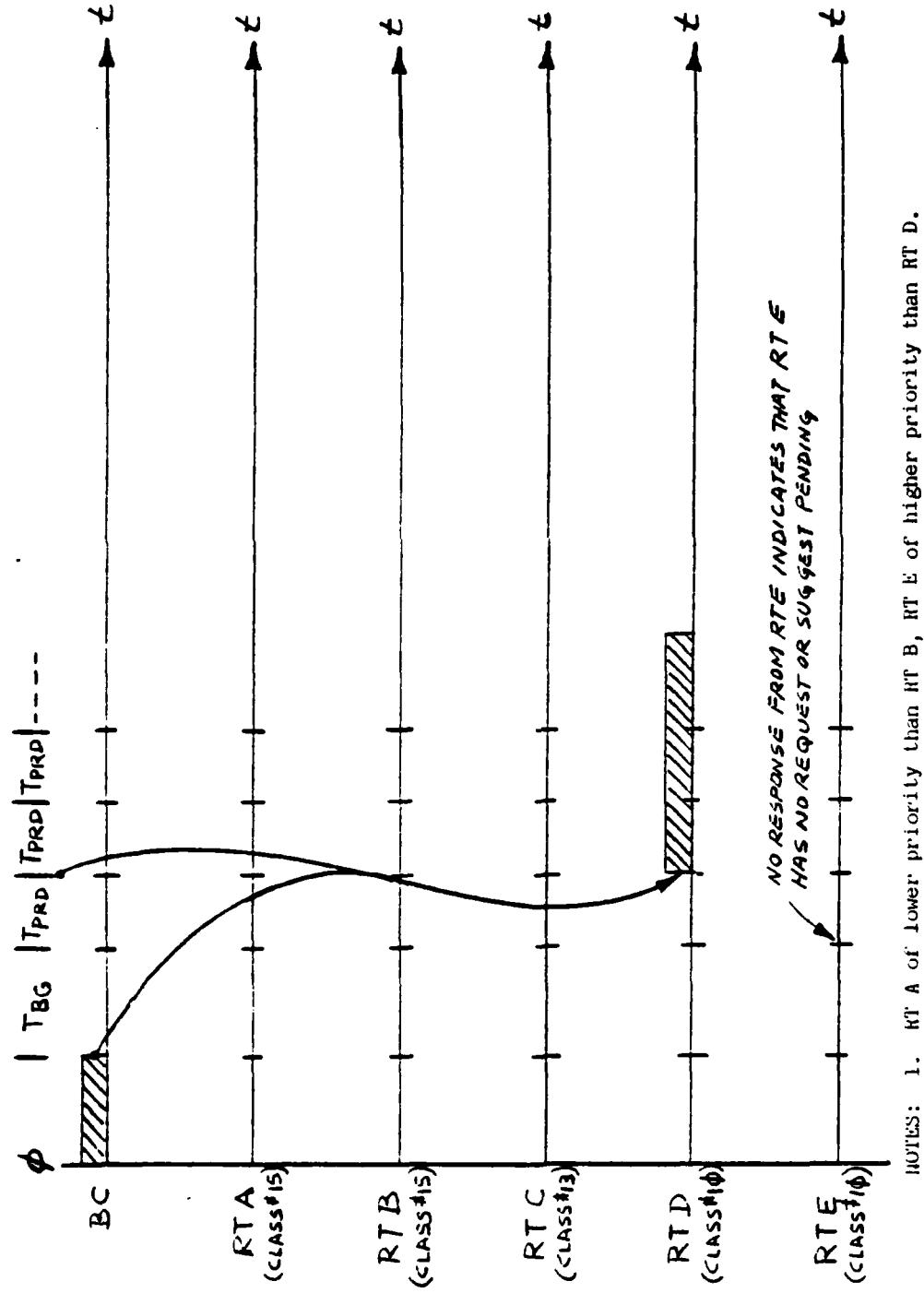
In the normal mode of operation (non-broadcast), Dynamic Bus Control Allocation is achieved by the BC first transmitting a Dynamic Bus Control Offer Mode Discrete. If the receiving bus unit wants control of the bus, it will respond with a Bus Control Acceptance status frame; however, the bus unit will not immediately take control of the bus but will begin timing out for a predetermined period in order that the current BC have enough time to process the received status frame and initiate any possible error recovery. If an error was detected in the status frame, the current BC must have sufficient time in which to either transmit a Retry Mode Discrete (after which the time out cycle shall be reinitialized by the prospective BC) or transmit an Unconditional Halt Mode Discrete. Only after a complete uninterrupted time out may the new bus controller assume control of the bus.

BIT #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	CW
	1	1	1	1	1	0	1	0	1	0	0	0	0	0	0	1	CW
						BC				bc		0	0	0	0	0	CW + 1 (ECW)
																	FCW

NOTES:

1. Response timing shown in Figure 3.3.4.3-1(b).
2. Class/Priority 10 arbitrarily set.
3. RT response (if any) will reflect a transmit or receive service request.
4. Broadcast poll shall only be executed if no transactions are currently active on the bus.

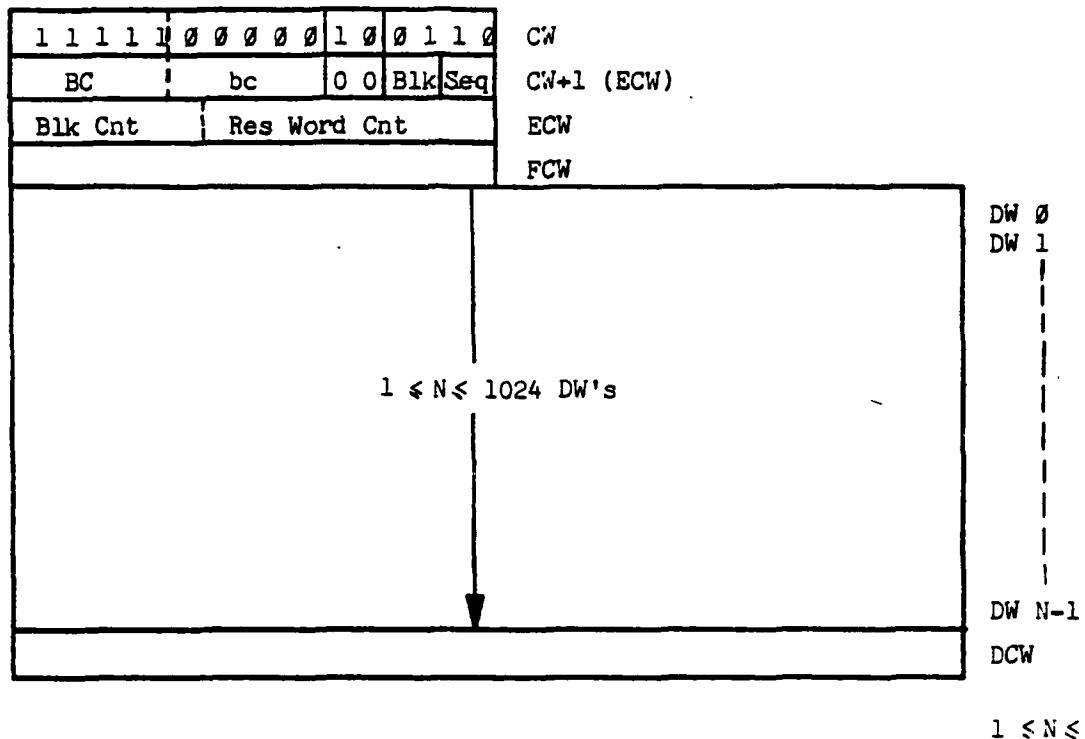
FIGURE 3.3.4.3-1(a) Broadcast Poll



NOTES:

1. RT A of lower priority than RT B, RT E of higher priority than RT D.
2. BC Broadcast Message from Figure 3.3.4.3(a).
3. Assume only 5 RT's on this bus.
4.  $T_{BG} - T_{PHD}$  in this example.

FIGURE 3.3.4.3-1(b) Broadcast Poll



Notes:

1. Class/priority  $\emptyset$  indicates all bus units accept data. Therefore, all bus units must be polled for status.
2. Multi-block transactions are implemented via repeated Broadcast Receive Suggest messages. Status polling after each block transfer.
3. See Figure 3.3.4-1(b) for further illustration of this example.

FIGURE 3.3.4.4-1(a) Broadcast RxS Message

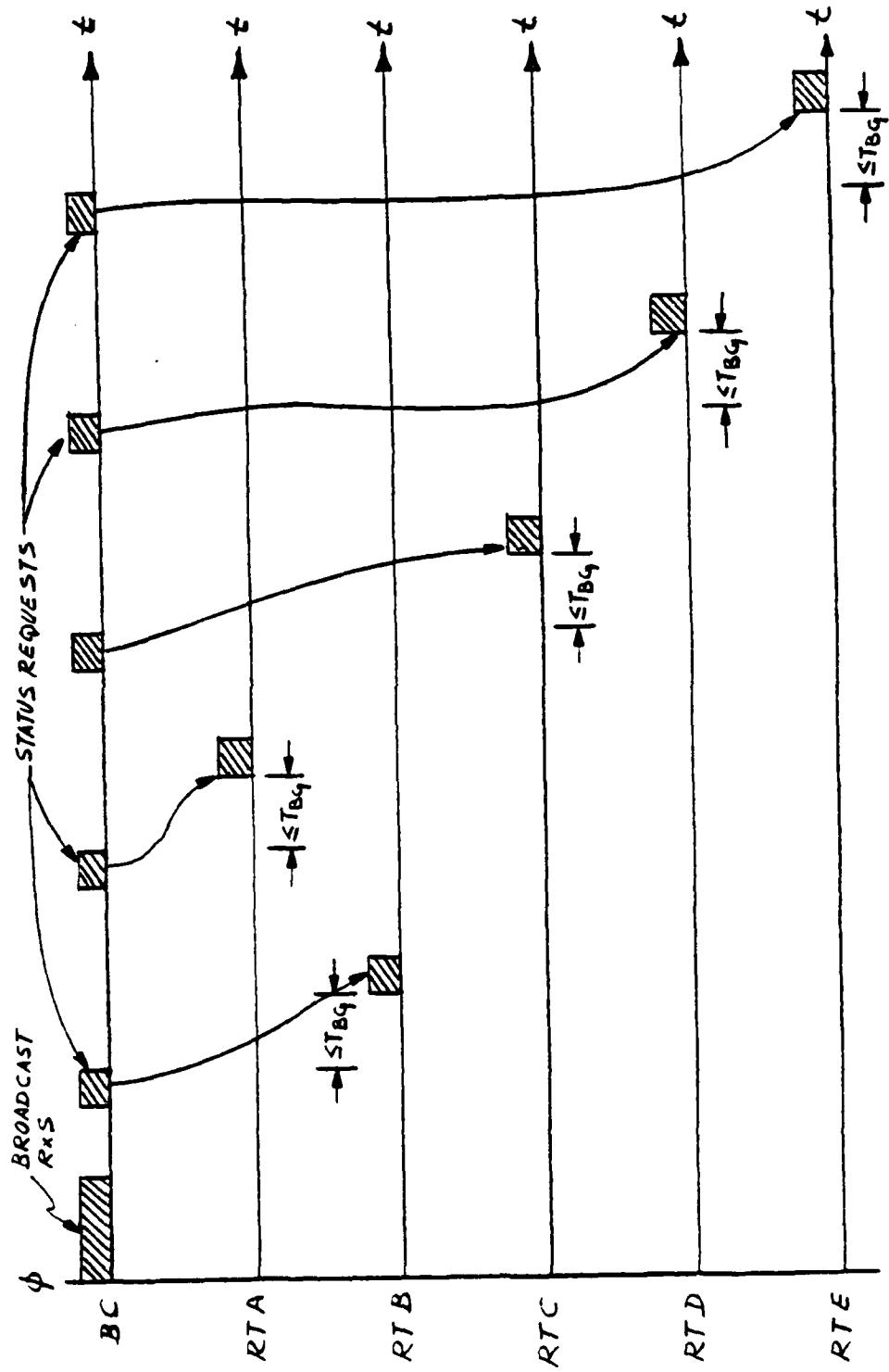


FIGURE 3.3.4.4-1(b) Broadcast RxS with Individual Status Polling

Broadcast operation for the Dynamic Bus Control Offer Mode Discrete shall proceed in the same manner as with the broadcast poll request. The first bus unit to respond with a Bus Control Acceptance status frame shall preclude any subsequent responses by any other bus units. The responding unit shall then follow the procedure described above in carrying out the transfer of bus control. It is implicit in this protocol that any bus unit which acquires control of the bus have the most updated set of bus control data concerning the subsystems which are attached to the bus. This update process may be accomplished in various ways, such as continuous bus monitoring, by potential bus controllers, periodic bus control data base updates of potential bus controllers by the current bus controller, or bus control data base transfers after bus control has been transferred, etc. The method to be used will be left to the system designer in order that specific advantage may be taken of particular architectural features of the target avionics system. Figure 3.3.4.5-1 gives an example of a dynamic bus control allocation message sequence.

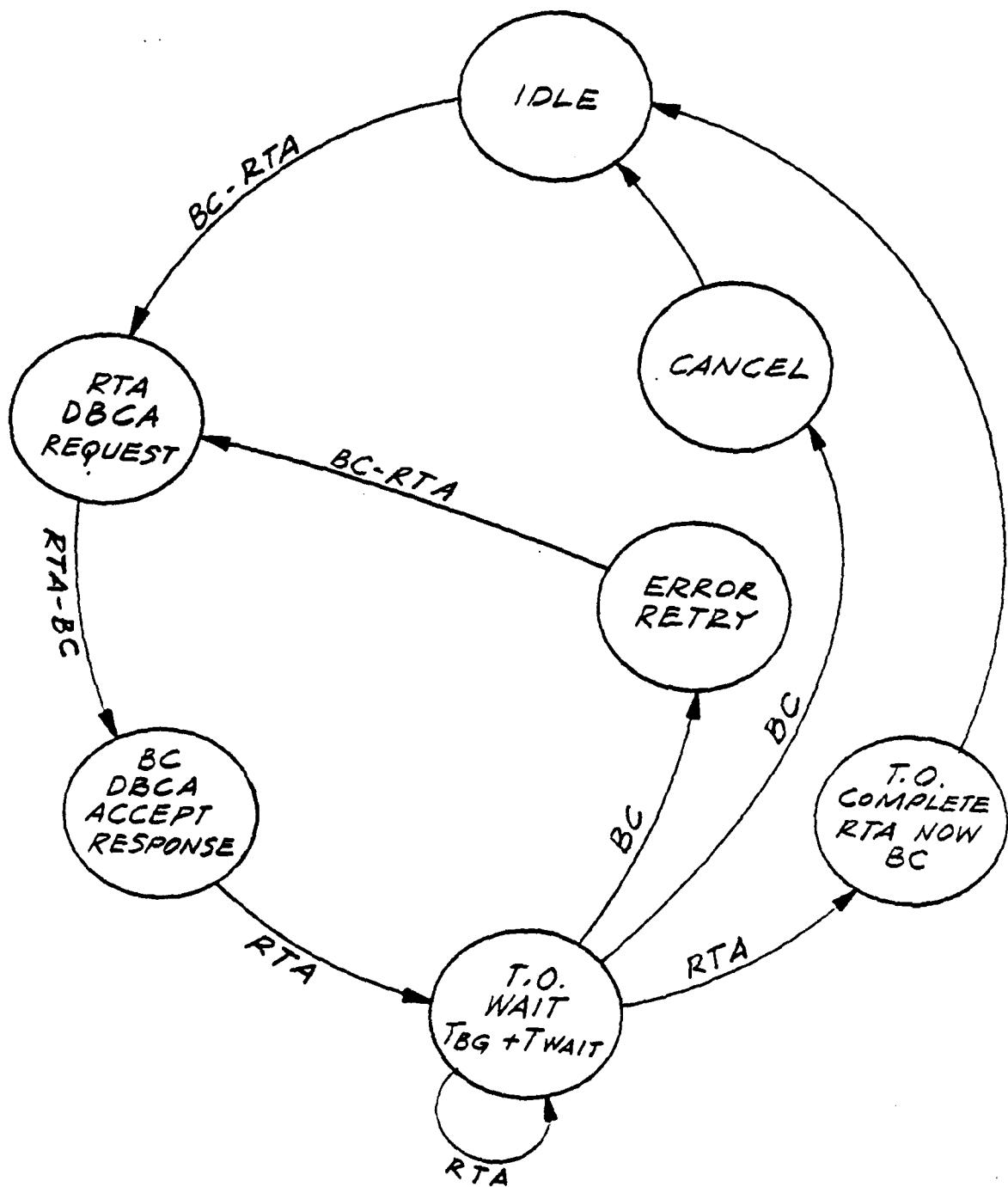
### 3.3.5 Error Modes

The error detecting features built into this protocol are classified into three main categories:

- a) Bit checking - Odd parity shall be implemented on a word basis.
- b) Context checking - The interpretation of the bit fields of the first word of both the command and status frames determines the exact number and utilization of words in the remainder of the command and status frames. The content of the command and status frame also generally reflects the content and context of any associated data in the transaction.
- c) Echo checking - In each transaction over the bus the status frame reflects the contents of the command frame thereby enabling the BC to double check, if necessary, that the intended bus unit received the intended message as the BC transmitted it.

The unique word synchronization patterns defined in section 4.5 also provide a degree of protection against random and burst errors via a combination of the above three techniques.

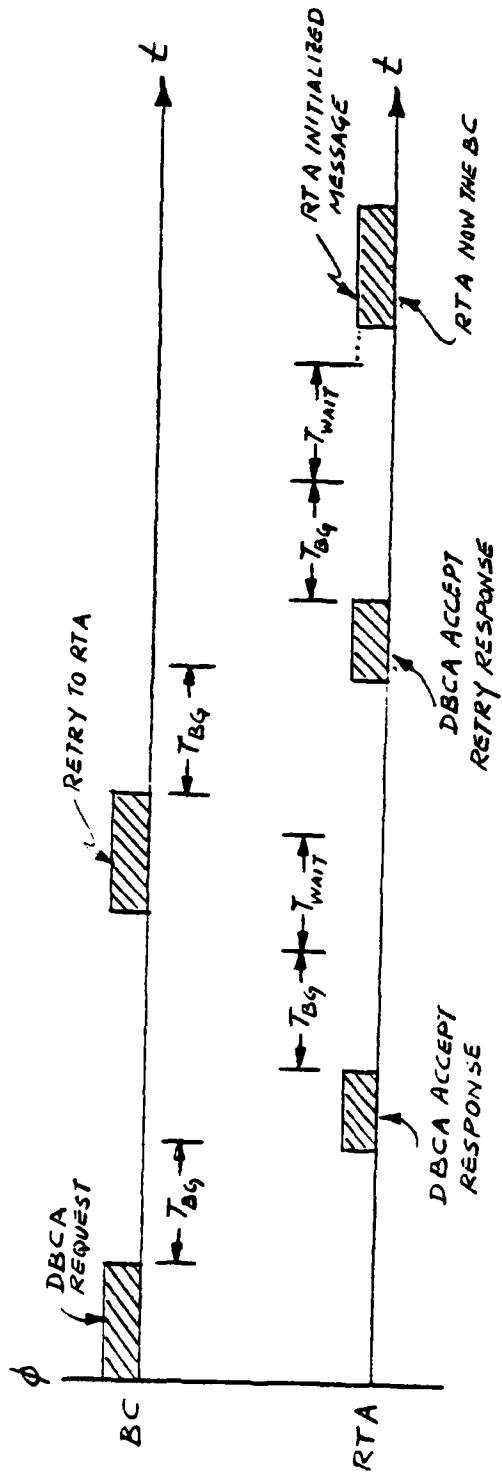
- d) Check Work Implementation - Check words are included as the last ECW in all command frames, the last ESW in all status frames, and as an appending DW to all data blocks. They are implemented to augment the word parity feature and thereby enhance the error detection capability of this protocol. The same algorithm shall be used to generate both the FCW's and the DCW and is (TBD). The FCW for the command frame shall be generated by the BC and verified by the RT independently of the DCW. Likewise, with the FCW in the status frame, it shall be generated by the RT and verified by the BC independently of the DCW. A check word error shall be treated as a message error in the same context as that of the parity error. Odd parity shall also be implemented for all check words.



NOTES:

1. Accepting RT must time out for  $T_{BG}$  plus an additional period,  $T_{Wait}$ , for BC to process the accepting RT's status.
2. Figure 3.3.4.5-1(b) illustrates the above diagram with one retry.

FIGURE 3.3.4.5-1(a) Dynamic Bus Control Allocation (DBCA) State Diagram



NOTES: 1. *Retry* due to possible Message Error or Protocol Error.  
2. *TWait* (TBD)

FIGURE 3.3.4.5-1(b) Time Referenced Dynamic Bus Control Allocation (DBCA) Sequence with A Retry

### 3.3.5.1 Error Types

As illustrated in figure 3.2.3.1-1, the reportable errors are separated into two types which are protocol errors and message errors.

#### 3.3.5.1.1 Protocol Error

A protocol error is defined to be the use of an existing field or sequence of words or messages for which no definition has been given or for which the particular system has not been implemented. A protocol error may also be caused by the improper operation of a subsystem. A receiving bus unit which detects a protocol error shall always respond with this status immediately (except after broadcast data transfers described previously). There are four protocol errors reportable under this protocol.

##### 3.3.5.1.1.1 Illegal Subaddress

An Illegal Subaddress error is caused by the specification of a subaddress which is not implemented in a particular RT or subsystem. In this situation the BIU (subaddress 0) shall respond with error status.

##### 3.3.5.1.1.2 Illegal Mode Discrete

This error is caused by the specification of Mode Discrete which is either defined to be illegal (CA = 5 or 14) or is not implemented in the particular bus unit.

##### 3.3.5.1.1.3 Subsystem Error

This error is caused by either a subsystem malfunction or by improper use of a subsystem or subsystem command. As this protocol is restricted to essentially the link level protocol, further error analysis must be accomplished through application software.

##### 3.3.5.1.1.4 Illegal Sequence

This error is caused by the unspecified use of ECW's or ESW's, the specification of transactions which are defined to be illegal, etc. For instance, the transmission by the BC of a broadcast Transmit Suggest is defined to be illegal (section 3.3.4) and shall cause an Illegal Sequence Error status to be generated and stored in the receiving bus unit.

#### 3.3.5.1.2 Message Error

A message error results from a violation of control or signal integrity and may or may not be immediately reported depending on where it is detected. With respect to figure 3.2.3.1-1, there are seven message errors (SA = 8-13, 15). Section 3.3.5.2 details when a bus unit may or may not respond with message error status.

#### 3.3.5.1.2.1 Word Count Error

A word count error will result whenever the number of DW's received or the number of blocks received does not match the number specified in the Word Count ECW (or ESW for the BC). Word count errors are always immediately reportable except following a broadcast Receive Suggest.

#### 3.3.5.1.2.2 Message Length Error

A message length error is detected whenever a received message lasts longer than the maximum allowable period for a full date block plus command (or status) frame. The maximum allowable message period is (TBD). This error is always immediately reportable except following a broadcast Receive Suggest.

#### 3.3.5.1.2.3 Sync Waveform/Bit Count Error

This error will be manifested by a received message with an illegal sync code, too many bits or too few bits in at least one word. There may be ambiguity at times in the determination of which of these conditions occurred; therefore, the three conditions are combined into one error code. Upon the occurrence of this error in the CW or any ECW of a command frame the receiving unit shall not immediately respond but will instead store the error status allowing the BC to time out. A Status Request Mode Discrete must be transmitted by the BC in order for the error status to be legally reported.

#### 3.3.5.1.2.4 Parity Error

A parity error is detected upon the receipt of any 16 bit command/status frame word or 32 bit DW which has an even number of 1's (including the parity bit). As with the Sync Waveform/Bit Count Error, should a parity error be detected in any word of the command frame, the receiving bus unit shall not immediately respond but shall instead store the error status allowing the BC to time out. A Status Request Mode Discrete must be transmitted by the BC in order for the error status to be legally reported. The sync identifier pattern for each word type (described in section 4.5) is not included in the parity determination defined above.

#### 3.3.5.1.2.5 FCW Error

An FCW error is detected whenever the check word generated by the receiving bus unit does not compare with the received FCW. Error processing and recovery for the FCW error shall be the same as for a parity error detected in a command or status frame.

#### 3.3.5.1.2.6 DCW Error

A DCW error is detected whenever the data check word generated by the receiving bus unit does not compare with the received DCW. Error processing

and recovery for the DCW error shall be the same as for a parity error detected in a data block.

### 3.3.5.1.2.7 Time Out Response Error

In a receiving bus unit this error condition is only legal in the RT-RT mode and then only with the receiving bus unit. Should this error code be used in any other situation it will itself constitute an illegal sequence error. The Time Out Response Error is caused by the transmitting bus unit in an RT-RT transfer not responding in the basic time out response period. The receiving bus unit, upon detecting this error, will generate and store the error status and wait for the BC to poll for status as defined in section 3.3.3.1.

### 3.3.5.2 Error Recovery

Error recovery is primarily implemented through the error detection and retry technique. Upon detection of an error as defined in the previous sections the BC shall retry the transaction a maximum of two times. If a transaction is unsuccessful on two consecutive retry attempts (three consecutive attempts including the original) then the BC shall perform the following steps:

- a) terminate the message sequence
- b) update the bus control data base
- c) inform the application software
- d) switch to the redundant bus and attempt to execute the transaction again (dependent on application software intervention).

Should all attempts to execute a transaction with a bus unit fail, the sequence shall be terminated and the application software shall be informed. It is the responsibility of the application software to initiate further action such as invocation of the Loop Test (section 3.4.6), elimination of the bus unit from the bus control data base, reconfiguration, etc. The error recovery steps outlined above shall be executed within the context of the target system bus timing cycles. Should a receiving bus unit/RT detect a message error in a command frame the error status shall be stored and the response to the BC shall be inhibited; however, should a message error be detected in a received data block then a status response shall be generated and transmitted to the BC indicating a message error. The BC shall then initiate error recovery.

## 3.4 Control and Information Exchange Sequences

The following message sequences make up the basic transaction types which are legal under this protocol. Specific examples of each message are presented in the referenced figures. Figure 3.4-1 and 3.4-2 show the general message types.

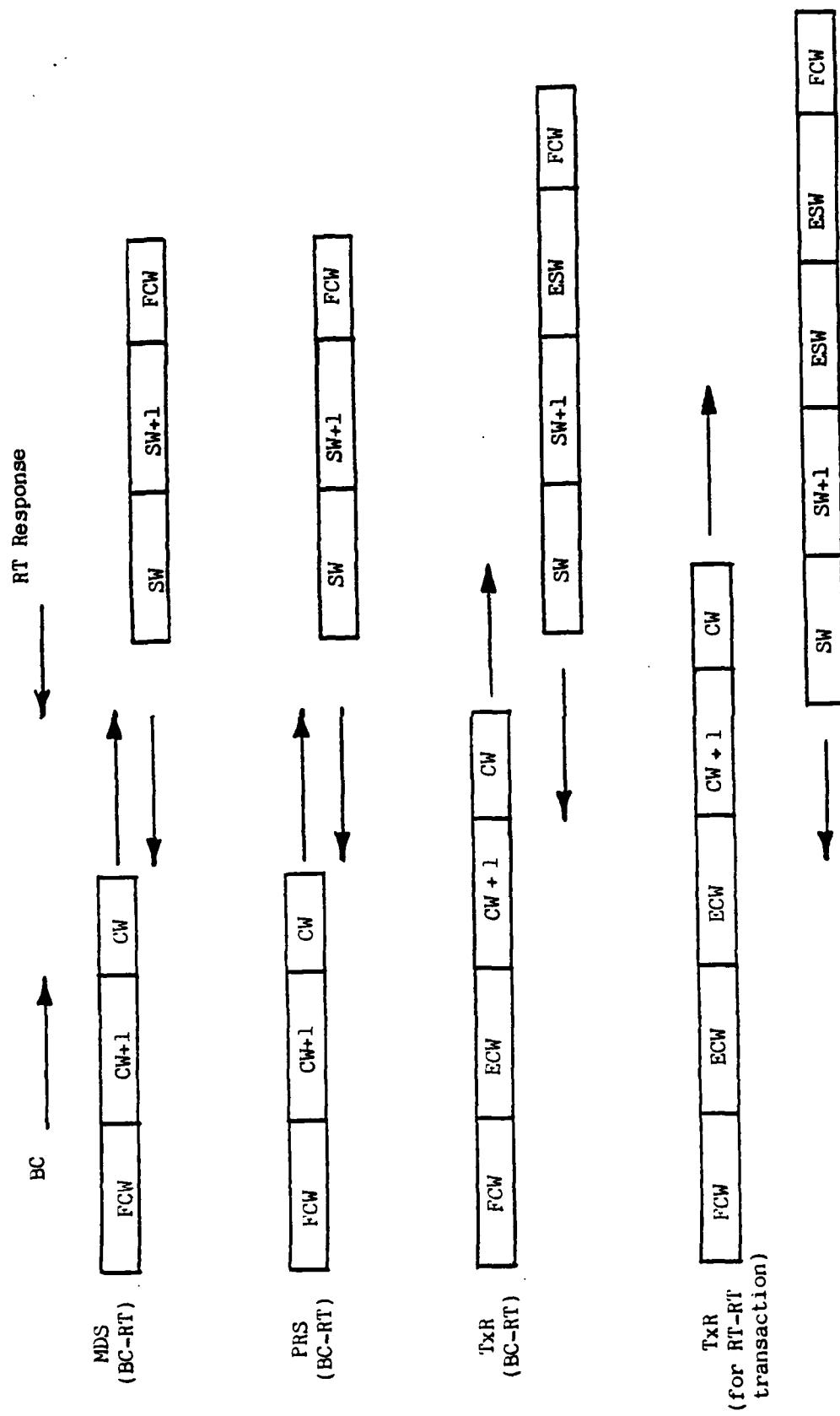


FIGURE 3.4-1 General Message Types

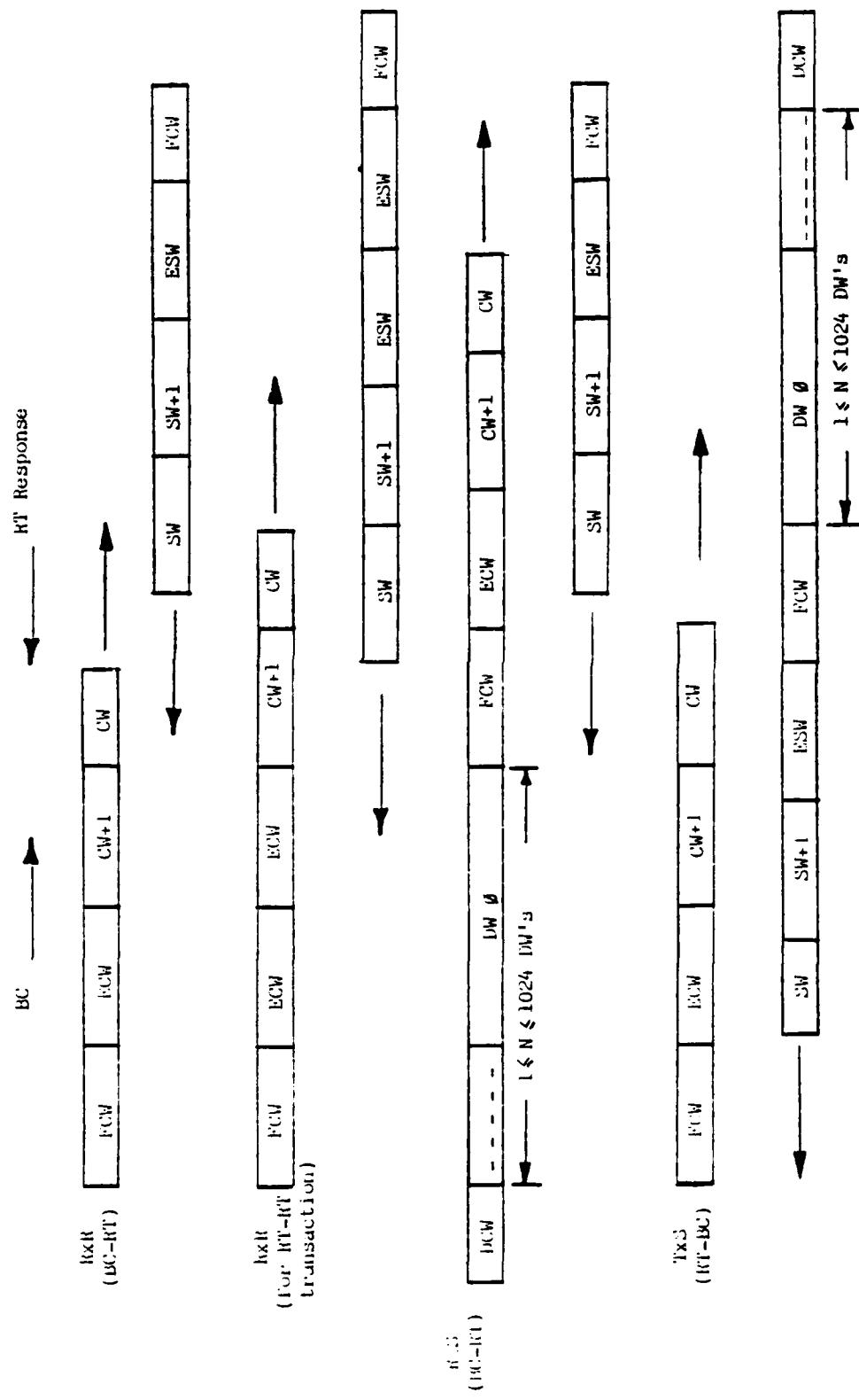


FIGURE 3.4-2 General Message Types

### 3.4.1 Mode Discrete Sequence (MDS)

An example of an MDS is given in figure 3.4-1 and described in section 3.1.19. This sequence is legal in the normal mode; however in the broadcast mode the Loop Test, Transmit BIT Message, Transmit Last Command Frame and Transmit Bus Control Data mode discretes are illegal.

### 3.4.2 Poll Request Sequence (PRS)

A typical PRS is given in figure 3.4-1. The PRS is legal in both the normal mode and the broadcast mode of operation. Figure 3.3.4.1-1 is a high level bus state transition diagram of a PRS.

### 3.4.3 Information Transfer Request (ITR)

This message type is illustrated in figures 3.4-1 and 3.4-2. There are two types of ITR; the Transmit Request (TxR) and the Receive Request (RxR). Figures 3.4.3-1 and 3.4.3-2 illustrate typical high level state transition diagrams of an ITR from the BC and RT points of view, respectively. The ITR diagrams in these two figures may be either Transmit Requests or Receive Requests.

#### 3.4.3.1 Transmit Request (TxR)

The basic TxR is shown in figures 3.4-1. This message is only legal in the normal mode of communication.

#### 3.4.3.2 Receive Request (RxR)

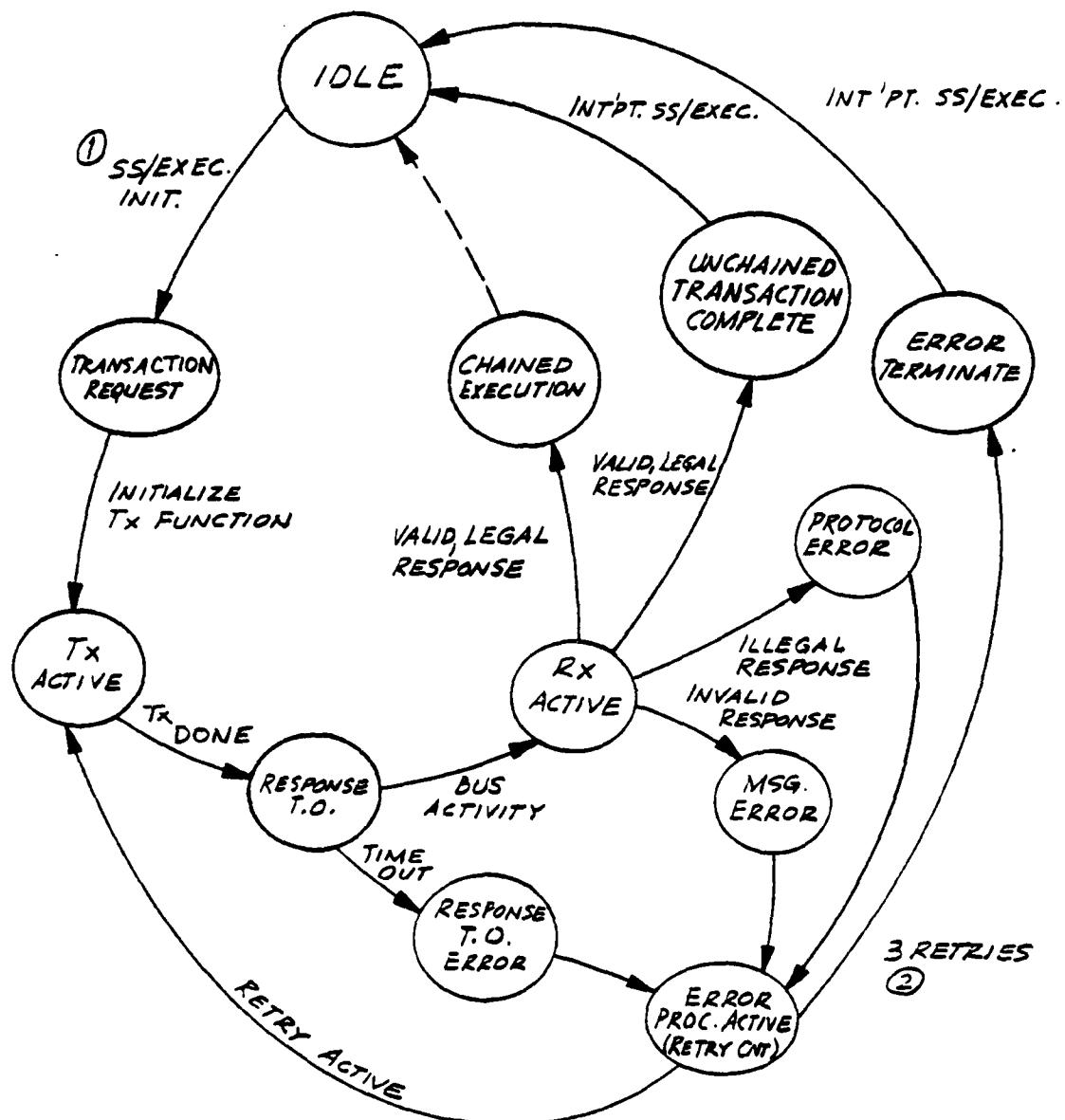
The basic RxR is shown in figure 3.4-2. This message is only legal in the normal mode of communication.

### 3.4.4 Information Transfer Suggest (ITS)

The ITS is the primary data transfer mechanism of this protocol and has two types; the Transmit Suggest and the Receive Suggest.

#### 3.4.4.1 Transmit Suggest (TxS)

The TxS is essentially a "Read Buffer" command to a bus unit with other message parameters specified in the command frame. An example of this transaction is given in figure 3.4-2. The TxS is legal only in the normal mode of communication.



(1) SS = Subsystem  
Exec = Executive

(2) Alternate bus transaction execution  
not shown (see section 3.3.5)

FIGURE 3.4.3-1 High level BC State Transition Diagram - BC-RT Transaction Request (Tx or Rx, Chai. - Unchained Request) with Error Transitions

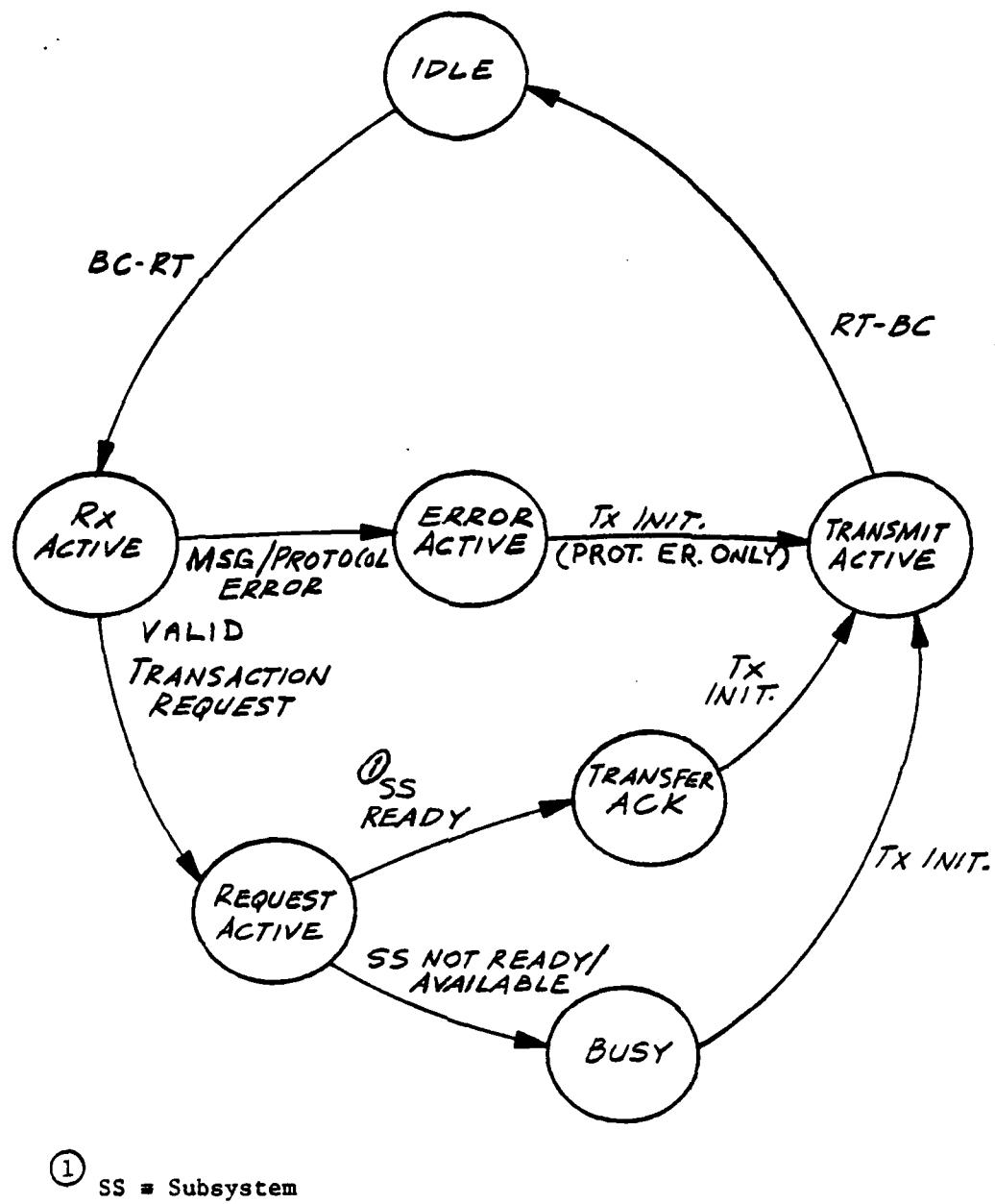


FIGURE 3.4.3-2 High Level RT State Transition Diagram - BC-RT Transfer Request (Tx or Rx Request) with Error Transition

#### 3.4.4.2 Receive Suggest (RxS)

The RxS is essentially a "Write Buffer" command to a bus unit with other message parameters specified in the command frame. An example of this transaction is given in figure 3.4-2. The RxS is legal in both the normal and broadcast modes of communication.

#### 3.4.5 Nesting

The nesting mechanism is defined in some detail in section 3.1.15. The inclusion of the nesting capability in a system or subsystem is a system design decision and is completely dependent on the communications requirements of the target application. However, should this capability be required the rules contained in this section shall apply.

The nest function description is broken into three areas - Bus Unit (RT) initiated, BC initiated and RT-RT nested transaction.

##### 3.4.5.1 Bus Unit (RT) Initiated Nest

During a multiblock transaction a nested request is initialized by the bus unit via the Service Request bit of the SA field (section 3.2.3.1). Upon detection of the Service Request bit being set and within the system cycle timing constraints the BC shall poll the bus unit. The bus unit (RT) shall then respond with a request with the nest field incremented by one (provided that it is not already at a value of 3). The BC shall then process the higher nested sequence in accordance with host software requirements. After the higher nested transaction is complete the BC shall then proceed with the next lower nest level transaction. Figures 3.4.5.1-1(a) and 3.4.5.1-1(b) give a high level state transition diagram of a nested group of transactions.

##### 3.4.5.2 BC Initiated Nest

It is also permissible that the BC initiate a nested transaction with a bus unit in the middle of a multiblock transaction. In this situation the BC must wait for the current message (of a multiblock, multimesage transaction) to successfully complete and then transmit a nested request to the bus unit with the nest field incremented by one (provided that the nest field is not already at a value of 3). The bus unit will respond to the nested request with either a "Busy" status or an acknowledgement (pending no errors) with the nest field incremented to the new value. The BC may then proceed with the new nested transaction, again in accordance with the system bus timing cycle. Figure 3.4.5.2-1 illustrates an example of a BC initiated nested transaction.

##### 3.4.5.3 RT-RT Nested Transactions

Nesting is relatively straight forward when the nested transactions involve only the BC and a bus unit. However, it is allowable under the protocol to also execute an RT-RT nested transaction as depicted by the diagrams of figure 3.4.5.3-1. In this situation an RT-RT nested transaction

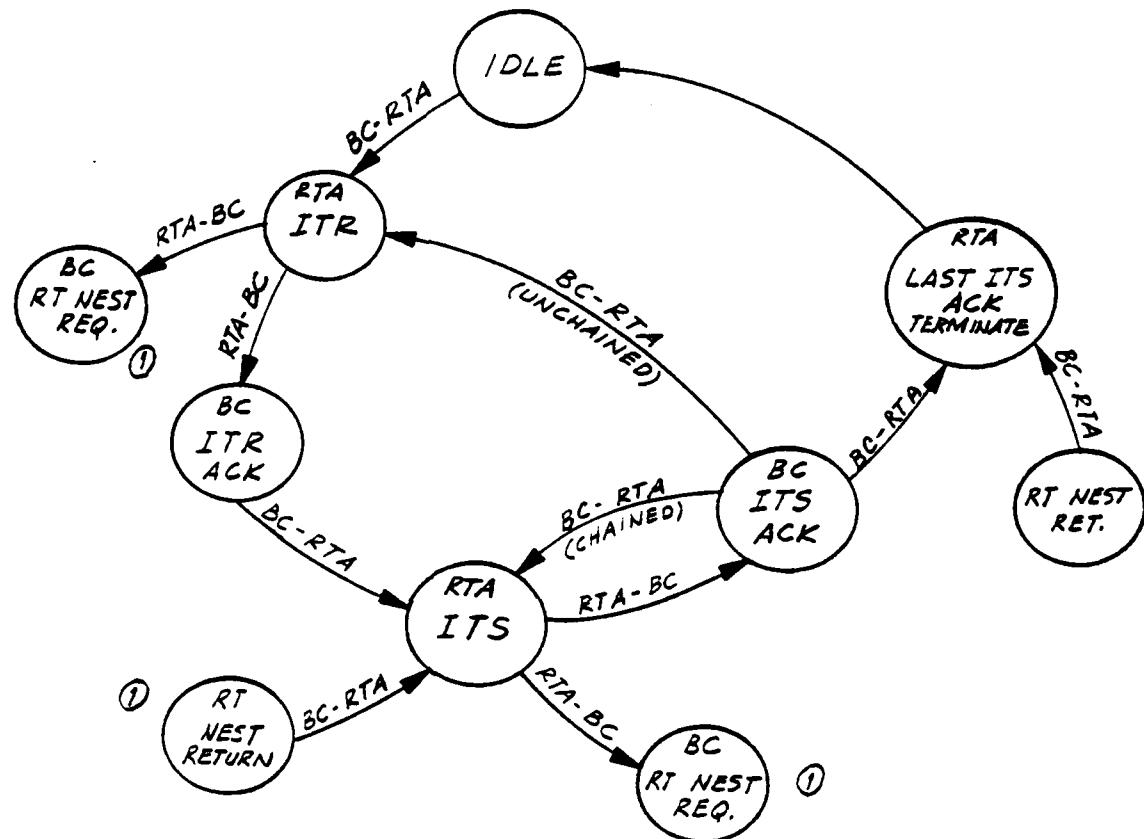
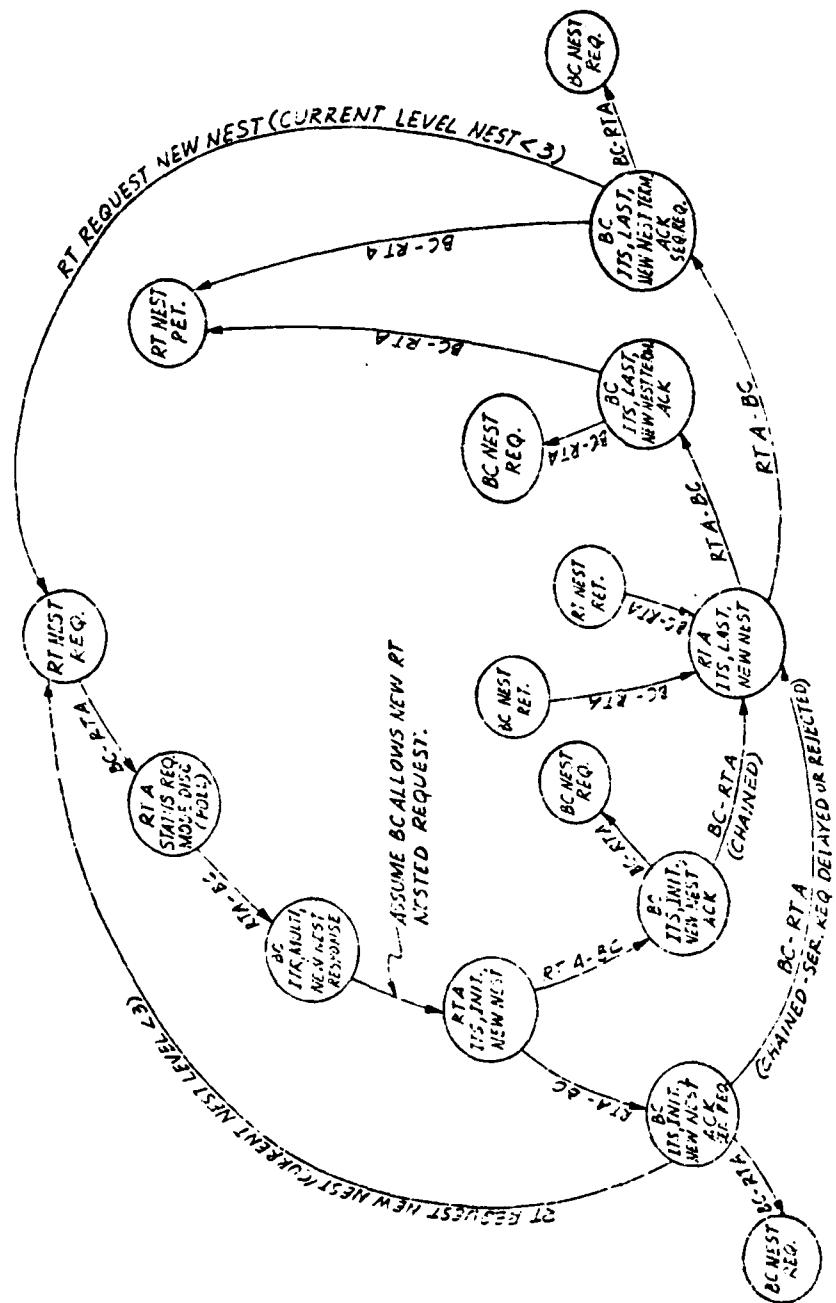


FIGURE 3.4.5.1-1(a) Nested Transaction - RT Initiated



- Shown here is a nested chained 2 block data transfer transaction
- Note that provisions exist for initiating another nested transaction while executing a nested transaction.
- The return point from a nested transaction to the suspended transaction dependent on the point in the first transaction at which the suspension occurred.

FIGURE 3.4.5.1-1(b) RT Initiated Nest State Diagram

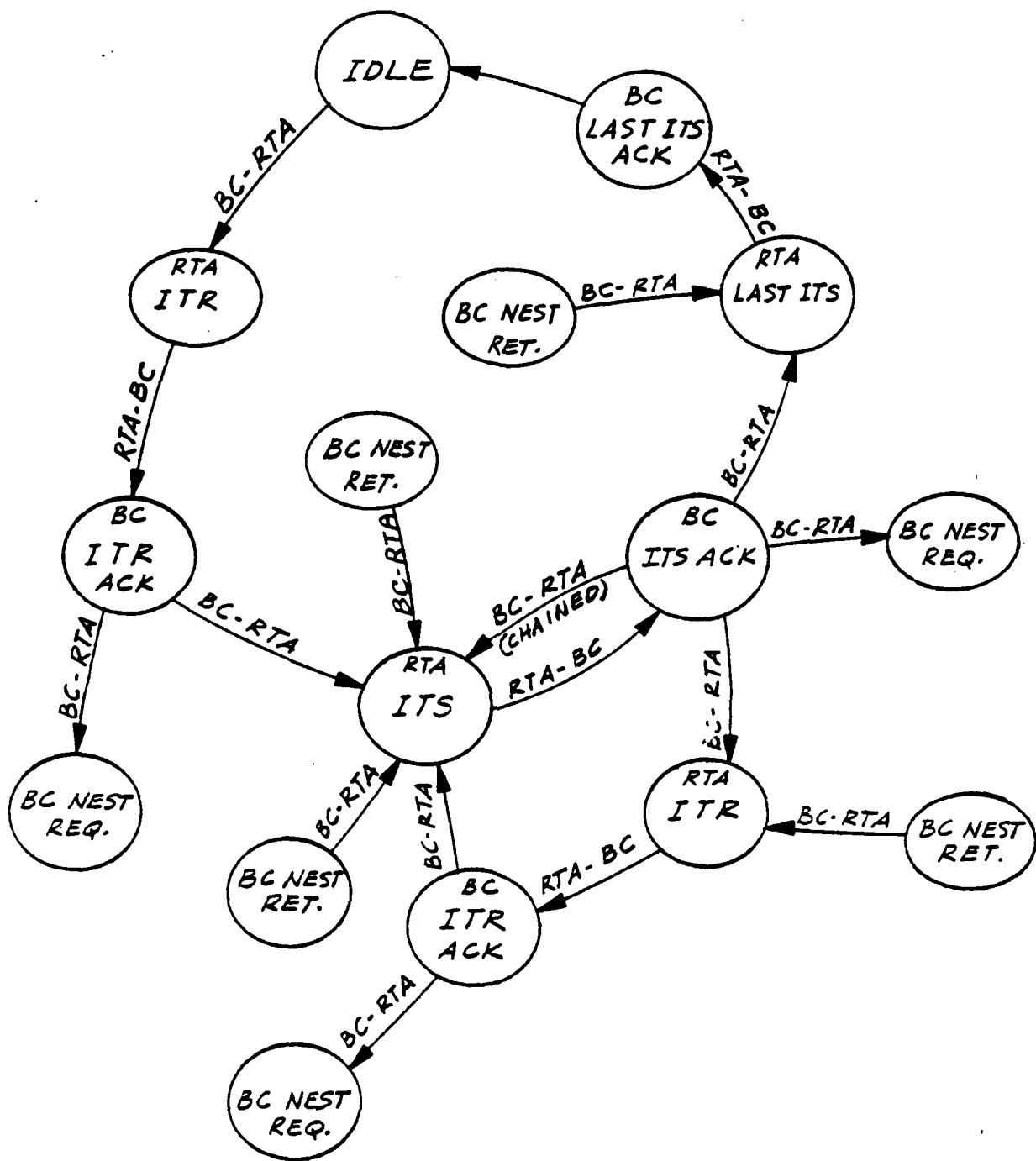
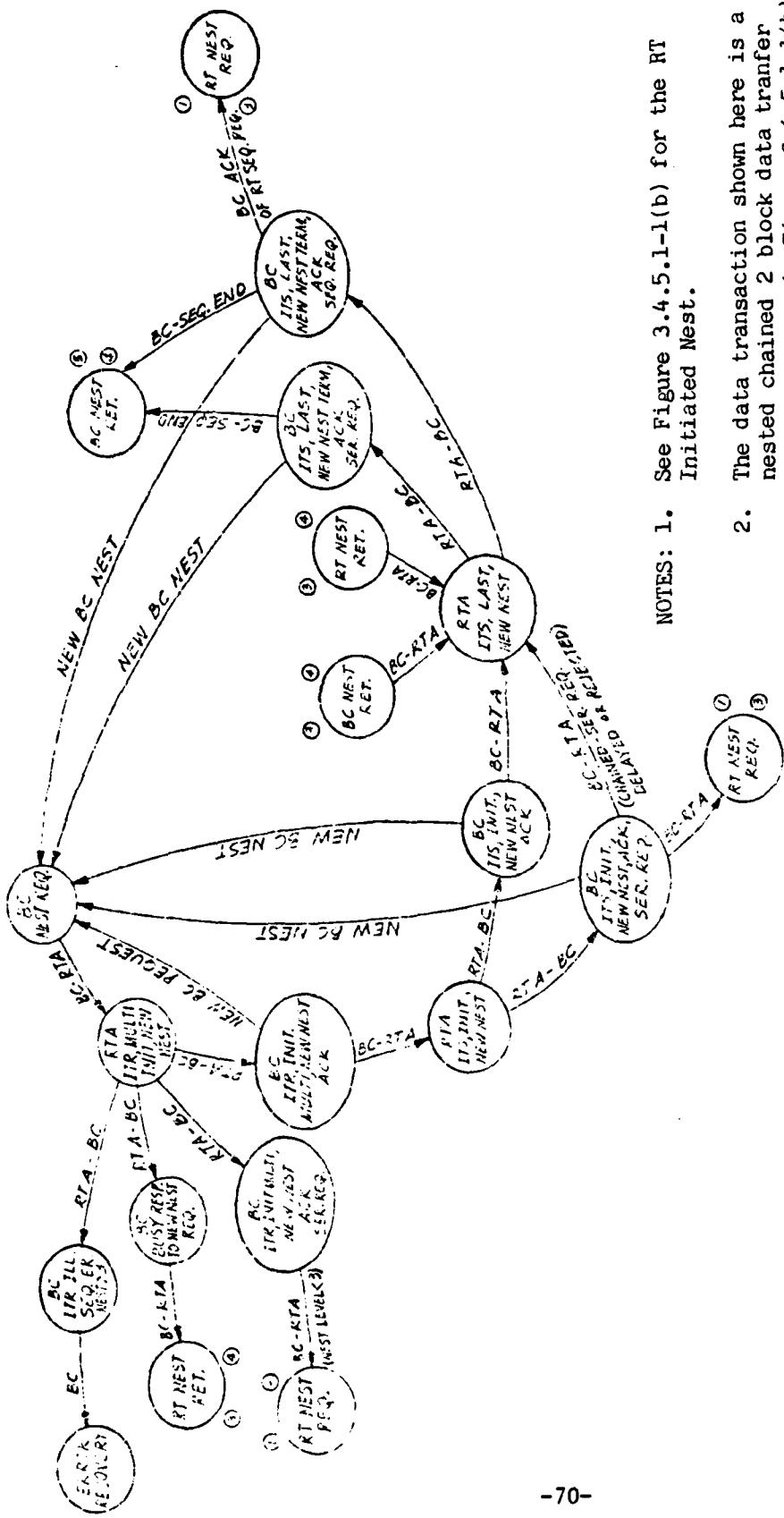


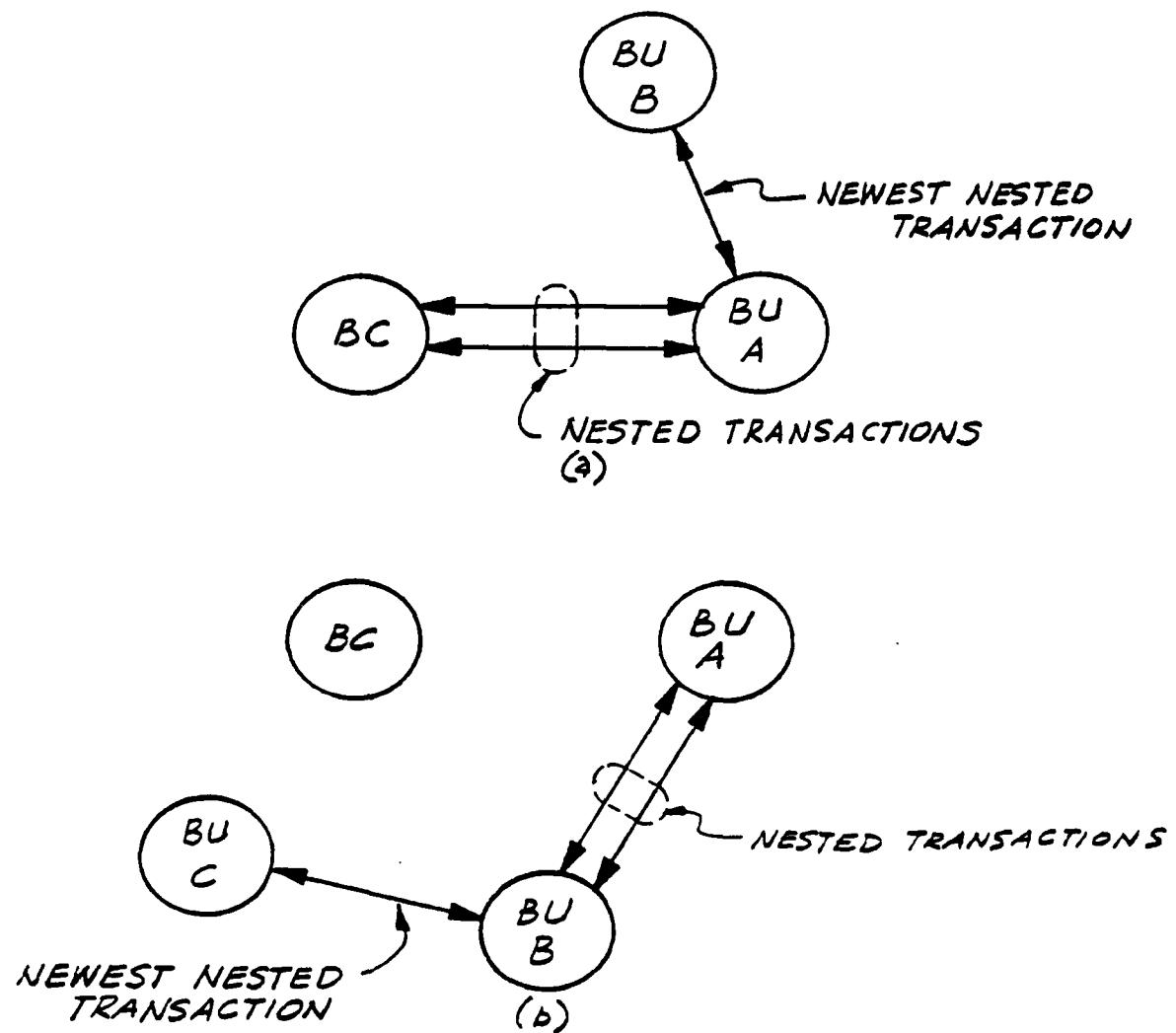
FIGURE 3.4.5.2-1(a) BC INITIATED NESTED TRANSACTIONS - HIGH LEVEL STATE DIAGRAM



NOTES: 1. See Figure 3.4.5.1-1(b) for the RT  
Initiated Nest.

3. Note that the capability is provided for multi-level nesting.
4. The return point from a nested transaction is dependent on the point in the first transaction at which the suspension occurred.

**FIGURE 3.4.5.2-1(b)** BC Initiated Nested Transaction - State Diagram



① Each vector indicates an active transaction with data flow in either direction.

FIGURE 3.4.5.3-1 NESTED TRANSACTIONS INVOLVING RT-RT TRANSFERS

may be initiated in a similar manner to the normal nested transaction described earlier. The actual nested RT-RT transaction, however, is executed according to the rules for RT-RT transactions defined in section 3.3.3. Nested RT-RT transactions must additionally meet the following two constraints:

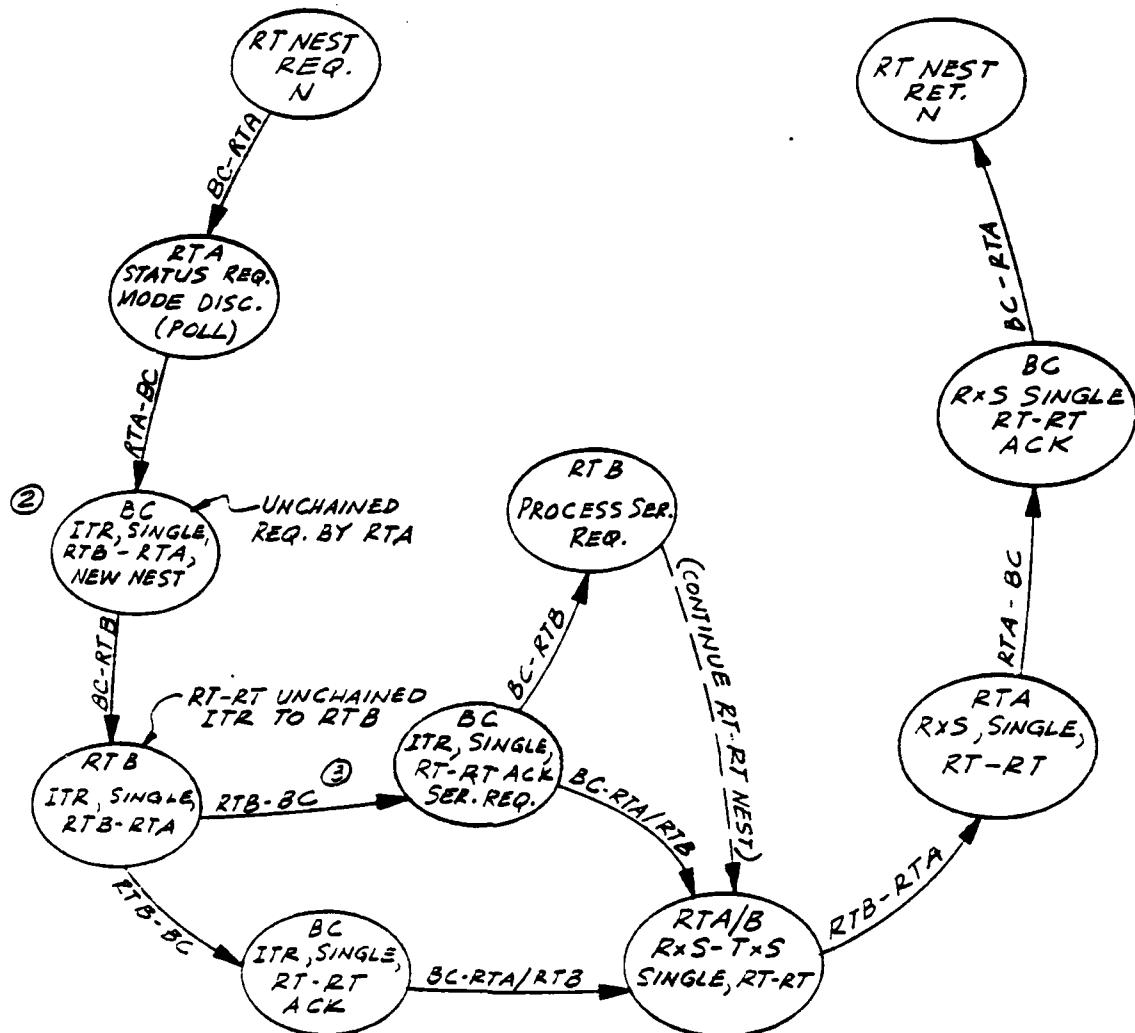
- a) The nest level of the newest transaction is specified by the BC and is acknowledged by the new bus unit. With respect to figures 3.4.5.3-1(a) and (b), B.U. B is the new bus unit in 3.4.5.3-1(a) and B.U. C is the new bus unit in 3.4.5.3-1(b).
- b) It is implicit in the protocol that a bus unit have sufficient buffering to be able to service all of the nested transactions for which it is implemented. Therefore, it is the responsibility of the BC that a nested transaction with a new bus unit ("new" as described in (a) above) will not exceed the capabilities of the new bus unit. Use of the bus control data base of the new bus unit by the BC and use of nested requests to the new bus unit by the BC can eliminate the possibility of this problem arising. Figure 3.4.5.3-2 gives an example of an RT-RT nested transaction.

### 3.4.6 Loop Test

The loop test function involves the execution of a predetermined set of message sequences between the BC and a bus unit. Following the successful completion of this set of message sequences, the loop test function will terminate automatically in both the BC and the designated bus unit and normal operation will resume.

#### 3.4.6.1 Loop Test Detailed Definition

(TBD)



NOTES:

1. Shown here is an RT-RT Unchained single block data transaction from RT B to RT A, initiated (requested) by RT A.
2. RT A requests unchained ITRs from RT B.
3. RT B may insert a Service Request in the ACK to the ITR. BC may or may not process it immediately (system dependent).
4. Error, Busy, and possible Nest Return and Nest Request (with exception of Note 3 above) states not shown.

FIGURE 3.4.5.3-2 RT-RT Nested Transaction - State Diagram

#### 4.0 GENERAL BUS CHARACTERISTICS

##### 4.1 Bus Architecture

The high speed multiplex bus shall be implemented in a dual redundant arrangement with a primary bus and a semi-passive redundant bus. The primary bus shall be designated Bus A with the redundant bus being designated Bus B. Bus B shall be the semi-passive backup bus in the sense that the BC shall periodically poll all bus units for status on this bus in order to ensure integrity and continuity of communications should a failure occur on Bus A. The polling period on Bus B shall be (TBD). Error processing in the event of errors during the polling cycle on Bus B shall be the same as those defined for the primary bus, Bus A, found in section 3.3.5. Unless Bus B is actually being used (in which case it becomes the primary bus) the status returned by the bus units in response to the polls shall be the Idle/MDS Response status except in the event of a detected error. As mentioned before, normal error processing will take place if errors are detected.

##### 4.2 Message Continuity

All words in a message shall be contiguous with the parity bit of one word followed immediately by the sync waveform of the next word. Due to possible ambiguities in the way this type of error would actually manifest itself, the status reported due to any gaps between words may be either Illegal Sequence Error or Sync Waveforms/Bit Count Error; however, this condition shall be detected as an error.

##### 4.3 Transmission Media

(TBD)

##### 4.4 Timing

###### 4.4.1 Response Times

(TBD)

###### 4.4.2 Message Length

(TBD)

###### 4.4.3 Intermassage Gap

(TBD)

#### 4.5 Word Synchronization Patterns

Design of the word synchronization patterns follows the philosophy described in section 3.3.5 (Error Modes). By the combination of bit checking, context checking and echo checking it is possible to detect any error or combination of errors in the sync patterns. Each word in a message is prefaced by a start waveform plus a three bit identification pattern. The three bit pattern is implemented as a 2-of-3 code in the following manner:

<u>Command Frame &amp; Data</u>		<u>Status Frame + Data</u>	
<u>2-of-3 Code</u>	<u>Meaning</u>	<u>2 of 3 Code</u>	<u>Meaning</u>
000	CW	111	SW
011	ECW	100	ESW
101	DW's	010	DW's
110	Check Words	001	Check Words

Note that for command frame words plus any associated data there is always a two bit difference in the identifier code. This is also true for status frame words plus any associated data. Note also that the identifier codes for command frame words and any associated data are the one's compliment of the corresponding status frame words and associated data. The start waveform for all words is an illegal Manchester waveform that is "logic 1" for 2 bit periods followed by a Manchester coded "0" pad bit. Figure 4.5-1 shows the sync patterns for all command frame words plus associated data words while figure 4.5-2 illustrates the sync patterns for all status frame words plus associated data words.

#### 4.6 Superceding Commands and Status

An RT shall accept a second message which is addressed to it and which meets the intermessage gap time of section 4.4.3. The second message, if valid, shall supercede a message which the RT is currently processing. The status frame shall always reflect the currently active sequence in the RT. Should a valid superceding message be received as described above then the status frame shall be updated to reflect the new message. The only exception to this is in the case of a valid status (poll) request in which case the status frame shall reflect the most recent state of the RT.

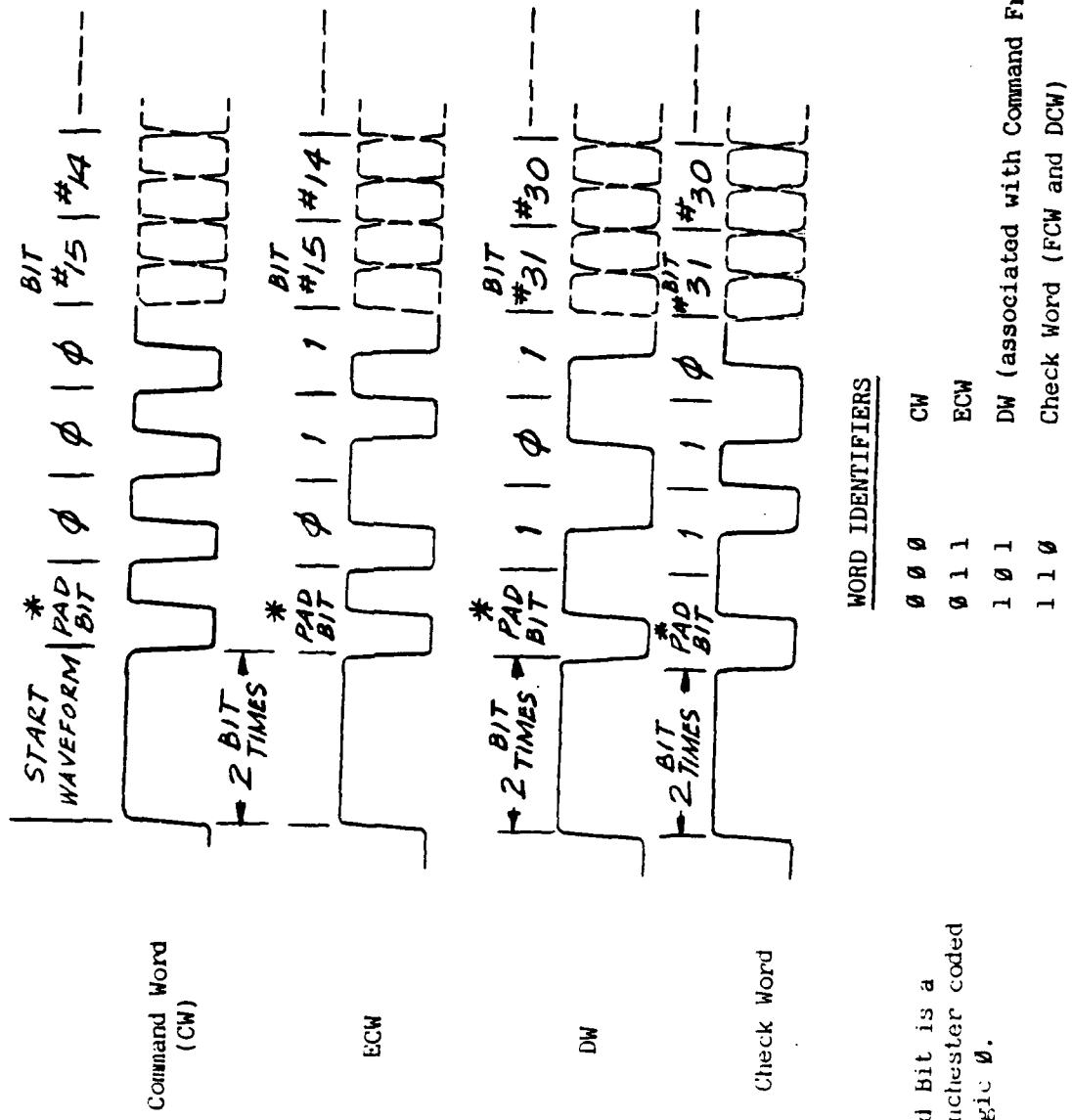


FIGURE 4.5-1 Command Frame Word Identifiers

\*\*Pad Bit is a Manchester coded logic 0.

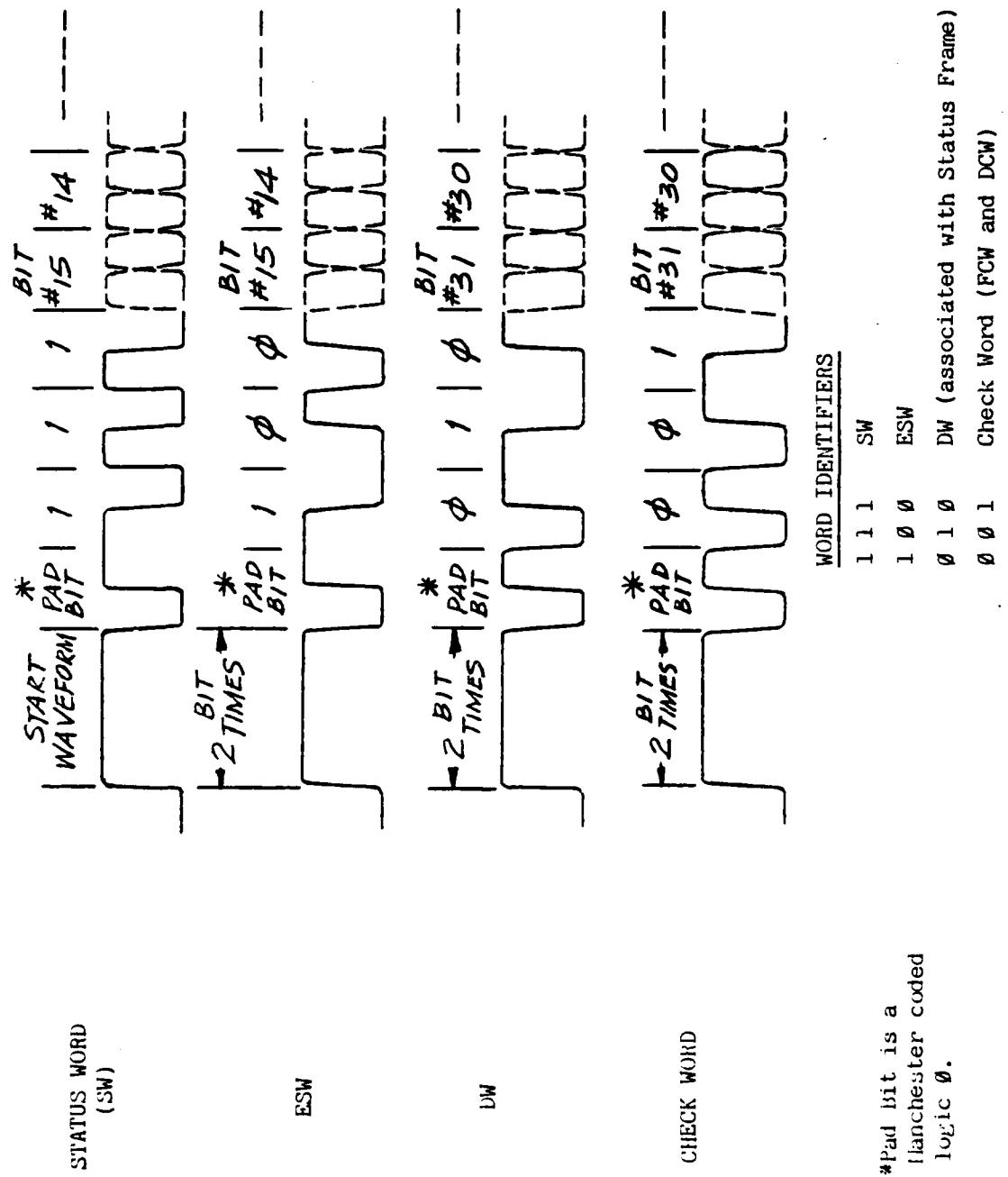


FIGURE 4.5-2 Status Frame Word Identifier

## 5.0 CONCLUSIONS

### 5.1 Application Considerations I

The thrust of this protocol is to provide the definition for as much of the actual communication control on the data bus as possible stopping short of and precluding the possibility of application or subsystem unique functions. As such it is incumbent on the system designer to optimize the implementation of the capabilities of this protocol for the target system without burdening the design with unnecessary functions. It shall be the responsibility of the system designer therefore, to only implement those functions which are required for the intended applications; however, within this protocol lies the inherent flexibility to extend the intersubsystem communications capabilities of the system in a consistent manner.

There are key points of commonality between this protocol and that of MIL-STD-1553A/B. These points include the address space, basic command word format (although some bits are rearranged and/or encoded) and basic transaction control strategy. This will facilitate, for any interface effort between the two busses, a mapping on a one to one basis in those areas of commonality. In addition, functionally, MIL-STD-1553 A/B is a subset of this protocol and this will also allow relative ease conceptually in any interface effort between these bus protocols. These points are extremely important from the implementation point of view (both hardware and software) for the following reason. Seldom, if ever, is an avionics suite developed and integrated using all new technology to the exclusion of older equipment. Instead, systems generally undergo a constant evolution of integration of new subsystems with older, in-place subsystems and there is no evidence or indication that this trend will not continue. Specifically, within the context of this development effort, MIL-STD-1553 is here today, its use is growing and it is likely to remain in avionics systems for some time to come. Any advanced bus and bus protocol which does not maintain some measure of compatibility with it (particularly in areas which will not degrade the performance of the new system) will only drive the complexity, development time and therefore cost of an interface effort between the two busses to an unnecessarily high level.

### 5.2 Application Considerations II

The preceding sections reflect a level of detail of protocol definition not found in MIL-STD-1553 A/B nor in other protocol descriptions targeted for general military avionics applications. In part, the parameters and rules documented in this protocol description are the result of considerations of various current military applications involving a multiplex bus based architecture plus considerations of future/planned avionics applications involving multiplex bus based architectures. One extremely obvious characteristic is common to all of the systems examined, regardless of the system architecture involved and this characteristic involves the mission orientation of an avionics suite coupled with the fact that a system

integrator (whether a military or industry prime) is responsible for defining the interdependence of all subsystems which will ensure successful execution of mission objectives. That is, in the context of this effort, an avionics system is considered to be a closed computer network. For the purposes of system definition, development, integration, and operational maintainability, it is under the complete control of one organization or entity. By the nature of its application, an avionics system must generally operate in a relatively isolated, generally stand-alone and geographically restricted environment (e.g., within the physical confines imposed by a fighter platform, ASW aircraft, etc.). The "nature of the beast" itself, as just described, encourages and imposes the need for (in many areas) a very high level of detailed definition from the start. Because subsystems within an avionics suite generally have independent, unique functions, the definition and interdependence which must be common to all subsystems must occur at the communication protocol level and the link level (the levels which are directly controlled by the BC and RT as illustrated in figure 2.1-1). It has been this concept which suggests that it may be feasible to go farther in protocol definition than has been done to date, and it is this concept which is emphasized by the preceding protocol definition.

NADC-81049-50

Bibliography

- 1) MIL-STD-1553B, Aircraft Internal Time Division Command/Response Multiplex Data Bus, 21 Sep. 1978.
- 2) MIL-STD-1553B Protocol Specifications for P-3B, document #1P-204-02, NADC, 15 Aug. 1979.
- 3) Applicability of MIL-STD-1553 Concepts to Interprocessor, High Speed Multiplex Data Bus; NADC, code 5022, 11 May 1979.
- 4) The Impact of Wideband Multiplex Concepts On Microprocessor-Based Avionics System Architectures, AFAL-TR-78-4; Jensen, Marshall, White, Helmbrecht; Feb 1978.
- 5) "Multiplex Applications Handbook"; final technical report prepared for AFSC, ASD-XRE, Wright Patterson AFB; Crossgrove, et al, Boeing Co. and Ellis, et al, SCI Systems Inc., 1 May 1979.

## ABBREVIATIONS and ACRONYMS

ACK	Acknowledge
BC	Bus Controller
BIU	Bus Interface Unit
BIT	Built In Test
B.U.	Bus Unit
CA	Command Amplifier
CC	Command Code
CDW	Control Data Word
CW	Command Word
DBCA	Dynamic Bus Control Allocation
DCW	Data Check Word
DW	Data Word
ECA	Extended Command Amplifier
ESA	Extended Status Amplifier
ESW	Extended Status Word
EXEC	Executive
FCW	Frame Check Word
IDW	Interrupt Data Word
ITR	Information Transfer Request
ITS	Information Transfer Suggest
LIFO	Last In First Out
MDS	Mode Discrete Sequence
ME	Message Error
NDW	Normal Data Word
NOOP	No Operation

PRS	Poll Request Sequence
Req.	Request
Ret.	Return
RT	Remote Terminal
RxR	Receive Request
RxS	Receive Suggest
SA	Status Amplifier
Seq.	Sequence
SC	Status Code
SS	Subsystem
SW	Status Word
T <sub>BG</sub>	Basic response gap time
T <sub>PRD</sub>	Priority response delay time
T <sub>WAIT</sub>	Wait time (dynamic bus control allocation)
TBD	To Be Determined
T.O.	Time Out
TxR	Transmit Request
TxS	Transmit Suggest